

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平9-205422

(43) 公開日 平成9年(1997) 8月5日

(51) Int.Cl. <sup>8</sup>	識別記号	庁内整理番号	F I	技術表示箇所
H 0 4 L 9/30			H 0 4 L 9/00	6 6 3 Z
G 0 9 C 1/00	6 2 0	7259-5 J	G 0 9 C 1/00	6 2 0 Z
	6 3 0	7259-5 J		6 3 0 D
		7259-5 J		6 3 0 F
H 0 4 L 9/08			H 0 4 L 9/00	6 0 1 D

審査請求 未請求 請求項の数18 OL (全 24 頁) 最終頁に続く

(21) 出願番号 特願平8-5277  
 (22) 出願日 平成8年(1996) 1月16日

(71) 出願人 390009531  
 インターナショナル・ビジネス・マシー  
 ズ・コーポレーション  
 INTERNATIONAL BUSIN  
 ESS MACHINES CORPO  
 RATION  
 アメリカ合衆国10504、ニューヨーク州  
 アーモンク (番地なし)  
 (72) 発明者 アミール・ヘルツベルグ  
 アメリカ合衆国ニューヨーク州、ブロンク  
 ス、ブラックストーン・アベニュー 3935  
 (74) 代理人 弁理士 合田 潔 (外2名)

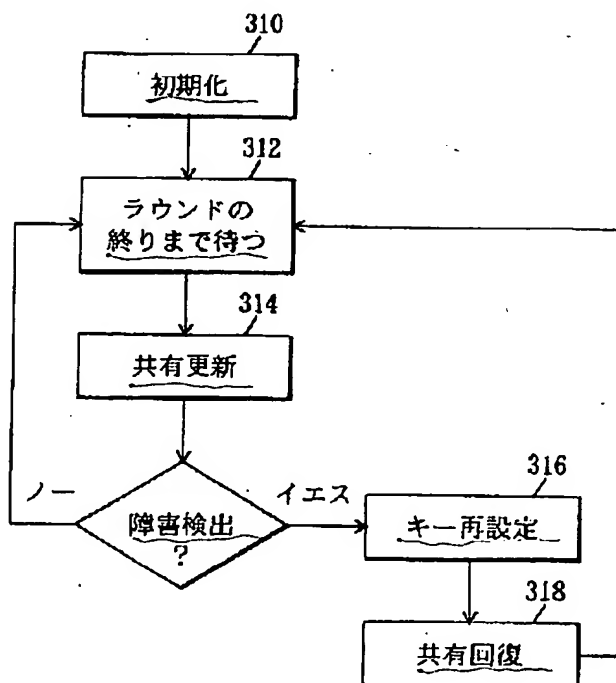
最終頁に続く

(54) 【発明の名称】 順向性、ロバスト及び回復可能な分散しきい値秘密共有を有する公開キー暗号システム及びその方法

(57) 【要約】

【課題】 順向性、ロバスト及び回復可能な分散しきい値秘密共有方式を有する公開キー暗号システム及びその方法の提供。

【解決手段】 順向性しきい値秘密共有暗号システムはサーバーのセットを用いる。nサーバーのうちの少なくとも(k+1)のサーバーが活動状態であり且つ正直である場合にサービスが維持される。秘密署名キーは、敵対者が少なくとも(k+1)サーバーに侵入した場合にのみコンプロマイズされる。また、正直なサーバーが誤りのあるサーバーを検出し、サービスは中断されない。さらに、敵対者がコンプロマイズしたサーバーにある全ての局所情報を敵対者が消去した場合でも、サーバーは正しいプロトコルの実行に復帰すると直ちに前記情報を復元できる。本発明の方法及びシステムは順向性を有するため、敵対者が秘密を知るためには、アルゴリズムの同じラウンドの間に(k+1)サーバーへの侵入が必要となる。



Best Available Copy

## 【特許請求の範囲】

【請求項1】 傾向性、ロバスト及び回復可能な分散しきい値秘密共有を有する公開キー暗号化の方法であって、キーを形成するために通信ネットワークによりリンクされたサーバーを初期化するステップと、終了を有する別個のラウンドで動作するように前記サーバーを同期させるステップと、前記通信ネットワークで同報通信されたメッセージから前記ラウンドの前記終了で更新されたキーを計算するステップと、コンプロマイズされたサーバーのセットを形成するために前記更新されたキーを検査するステップと、前記コンプロマイズされたサーバーのセットを回復するステップとを含む方法。

【請求項2】 前記初期化するステップは、前記サーバーの各々の乱数を選択するステップと、前記乱数から前記サーバーの各々の秘密値を計算するステップと、前記秘密値から前記サーバーの各々の秘密キーを計算するステップと、前記秘密キーの公開対応部を前記通信ネットワークで同報通信するステップとを含む、請求項1に記載の方法。

【請求項3】 前記更新されたキーを計算するステップは、前記サーバーの各々の多項式を定義する乱数のセットを取出すステップと、前記多項式から導き出された、前記サーバーの各々の新しい秘密キーを取出すステップと、前記多項式から導き出されたメッセージを前記通信ネットワークで同報通信するステップとを含む、請求項1に記載の方法。

【請求項4】 前記更新されたキーを検査する前記ステップは、無効なサーバーのセットを形成するために前記通信ネットワークで同報通信された前記メッセージを分析するステップと、前記無効なサーバーのセットに対応する告発のセットを生成するステップと、前記告発のセットを前記通信ネットワークで同報通信するステップと、更新された無効なサーバーのセットを形成する前記同報通信された告発のセットを分析するステップとを含む、請求項1に記載の方法。

【請求項5】 前記コンプロマイズされたサーバーのセットを識別するメッセージをコンソールに表示するステップを更に含む、請求項1に記載の方法。

【請求項6】 前記サーバーのセットを回復するステップは、前記コンプロマイズされたサーバーのセットにある各サーバーに新しい秘密キーを設定するステップと、

前記サーバーから回復サーバーのセットを選択するステップと、前記回復サーバーのセットにある各サーバーの下位共有を計算するステップと、前記下位共有から取出されたメッセージを前記通信ネットワークで同報通信するステップと、前記コンプロマイズされたサーバーのセットが受取った前記下位共有から取出された前記メッセージを検査するステップとを含む、請求項1に記載の方法。

【請求項7】 回復サーバーのセットを選択する前記ステップは、前記サーバーの下位セットを選択するステップと、前記下位セットの各サーバーの乱数値のセットを計算するステップと、前記乱数値のセットから取出された署名されたメッセージを前記通信ネットワークで同報通信するステップと、無効サーバーのセットを取出すために前記署名されたメッセージを検査するステップと、前記サーバーの前記下位セットから前記無効サーバーのセットを除去するステップとを含む、請求項6に記載の方法。

【請求項8】 前記サーバーの各々の乱数を取出すステップと、前記乱数から前記サーバー毎の第1の署名の部分を計算するステップと、前記第1の署名の部分を前記通信ネットワークで同報通信するステップと、前記第1の署名の部分から前記サーバーの各々の第2の署名の部分を計算するステップとを更に含む、請求項1に記載の方法。

【請求項9】 署名されたメッセージを前記第2の署名の部分により検査するステップを更に含む、請求項8に記載の方法。

## システム・クレーム

【請求項10】 傾向性、ロバスト及び回復可能な分散しきい値秘密共有を有する公開キー暗号方式を処理するデータ処理システムであって、通信ネットワークによりリンクされたサーバーと、前記サーバーと関連したキーを形成するために前記サーバーを初期化する初期化手段と、終了を有する別個のラウンドに前記サーバーの動作を同期させるタイミング手段と、更新されたキーを生成するために前記別個のラウンドの各ラウンドの終了で前記キーを更新する更新手段と、コンプロマイズされたサーバーのセットを形成するために前記更新されたキーを検査する検査手段と、前記コンプロマイズされたサーバーのセットを回復する回復手段とを備えるデータ処理システム。

【請求項11】 前記サーバーの各々の乱数を選択する選択手段と、前記乱数から前記サーバーの各々の秘密値を計算する第

1の計算手段と、

前記秘密値から前記サーバーの各々の秘密キー及び前記秘密キーの公開対応部を計算する第2の計算手段と、  
 前記秘密キーの各々の前記公開対応部を前記通信ネットワークで同報通信する同報通信手段とを備える、請求項10に記載のデータ処理システム。

【請求項12】前記更新手段は、

前記サーバーの各々に関連した、多項式を定義する乱数のセットを生成する乱数生成手段と、  
 前記多項式から前記サーバーの各々の新しい秘密キーを生成する秘密キー生成手段と、  
 前記通信ネットワークで前記多項式から取出されたメッセージを同報通信する同報通信手段とを備える、請求項10に記載のデータ処理システム。↓

【請求項13】前記検査手段は、

無効なサーバーのセットを形成するために前記通信ネットワークで同報通信されたメッセージを分析する第1の分析手段と、  
 前記無効なサーバーのセットに対応する告発のセットを生成する告発手段と、  
 前記告発のセットを前記通信ネットワークで同報通信する同報通信手段と、  
 無効なサーバーの更新されたセットを形成するために前記同報通信された告発のセットを分析する第2の分析手段とを備える、請求項10に記載のデータ処理システム。↓

【請求項14】前記コンプロマイズされたサーバーのセットを表示する表示手段を更に備える、請求項10に記載のデータ処理システム。

【請求項15】前記回復手段は、

前記サーバーから回復サーバーのセットを選択する選択手段と、  
 前記回復サーバーのセットにあるサーバーの各々の下位共有を計算する計算手段と、  
 前記回復サーバーのセットにあるサーバーの各々の前記下位共有から取出されたメッセージを前記通信ネットワークで同報通信する同報通信手段と、  
 前記コンプロマイズされたサーバーのセットが受取った前記下位共有の各々から取出された前記メッセージを検査する検査手段とを更に備える、請求項10に記載のデータ処理システム。↓

【請求項16】前記選択手段は、

前記サーバーの下位セットを選択する選択手段と、  
 前記下位セットにある各サーバーの乱数値のセットを計算する計算手段と、  
 前記乱数値のセットから取出された署名されたメッセージを前記通信ネットワークで同報通信する同報通信手段と、  
 無効なサーバーのセットを取出すために前記署名されたメッセージを検査する検査手段と、

前記無効なサーバーのセットを前記下位セットから除去する除去手段とを備える、請求項15に記載のデータ処理システム。

【請求項17】前記サーバーの各々の乱数のセットを生成する乱数生成手段と、

前記乱数のセットから前記サーバーの各々の署名の第1の部分を計算する第1の計算手段と、

前記署名の第1部分を前記通信ネットワークで同報通信する同報通信手段と、

前記第1の署名の部分から前記サーバーの各々の署名の第2の部分を計算する第2の計算手段とを更に備える、請求項10に記載のデータ処理システム。↓

【請求項18】署名されたメッセージを前記第2の署名の部分により検査する検査手段を更に備える、請求項17に記載データ処理手段。↓

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明はデータ処理システムに、より詳しくは情報保全のための暗号機構を備えるデータ処理システムに関する。

【0002】

【従来の技術】暗号化されたメッセージを送受するために公開キー暗号システムが用いられる。公開キー暗号システムは、公開キーと呼ばれる公に入手できる値を用いる数理的なアルゴリズムの実行によりメッセージが暗号化されるシステムである。そして、受取人は秘密キーと呼ばれる専用の値を用いるアルゴリズムを実行することによりメッセージを解読する。公開キー暗号は暗号アルゴリズムE及び解読アルゴリズムDの選択に依存し、Eが完全に解読されても、Dの取出しは実効的に不可能であるようにする。公開キー暗号の3つの必要条件は：

(1)  $D(E(P)) = P$ 、ここで、Pはメッセージである；

(2) EからDを演繹することは極めて困難である；及び

(3) Pを攻撃してもEを破壊できない↓

ことである。従って、公開キーは自由に配布できるが、秘密キーは、それを用いるエンティティにより秘密が保持されなければならない。侵入者がエンティティのメモリ内容をアクセスした場合、システム保全は侵害される。これは最初の公開キー特徴の全てにあてはまる。キー第三者委託システム及び「ミカリ(Micali)」の規則にかなった暗号システムでは、秘密キーは多くの部分に分割され、そして各部分は異なるエンティティにより保持される。しかしながら、時間の経過とともに侵入者が各エンティティのメモリを読み取り得る場合、システム保全は侵害される。↓

【0003】【傾向性秘密共有】傾向性秘密共有の概念を説明するために、秘密共有方式の用語及びそれらの保全特性を説明する。

【0004】【秘密共有におけるしきい値、ロバスト

(堅固性) 及び回復可能性] 秘密共有は最初に G.R. Blakey, Safeguarding Cryptographic Keys, AFIPSCon. Proc. (v.48), 1979, pp. 313-317及び A. Shamir, How to Share a Secret, Commun. ACM, 22, 1979, pp. 612-613 により導入された。|その最も基本的な形式では、秘密共有は、 $n$ 人の参加者の間で秘密情報 $M$ を分割し、参加者らが互いにそれを再構築できるようにする方法である。|しかしながら、 $(n-1)$ の共有保持者のグループはどれも $M$ に関して何も知ることはできない。この機構は個々の情報を守る必要があるときは保全を高めるために必ず用いられる。保全を高めることは、サーバーの全てではないが、その幾つかに侵入できる敵対者から秘密を守りうることに相当する。あらゆる秘密共有方式は概念的に以下の段階を有する。|

- (1) 秘密を知っているディーラーがその共有を生成し、それらを共有保持者に配布するディーリング段階;
- (2) 共有保持者の間で共有が維持される記憶段階;及び
- (3) 共有保持者がかれらの共有から秘密を再構築する再構築段階 |

上記の基本的な秘密共有はあまり実際的ではない。なぜなら、敵対者が共有保持者をコンプロマイズし、当該保持者が保持した秘密共有を消去又は変更する場合、秘密 $M$ は決して再構築できないからである。|よって、上記の方式は、 $(n-1)$ サーバーに侵入してそれらの共有を知りうるが、サーバーのメモリ即ち記憶を消去又は変更することも、サーバーにクラッシュさせることもできず、そして共有の処理にも秘密の再構築にも干渉しない敵対者に対して安全である。|

【0005】基本的な方式の保全性を高める秘密共有の特性が定義される前に、サーバーに対する敵対者の異なるタイプの攻撃が定義されなければならない。|

・敵対者がサーバーに侵入し、そのサーバーに記憶された全ての秘密情報：その秘密共有及び通信に用いるキーを知る場合、サーバーは敵対者にコンプロマイズされる。

・敵対者がサーバーに仕事を止めさせる場合、敵対者はサーバーを凍結する。しかしながら、敵対者が追放されると直ちに、サーバーは正しいプロトコルの実行に復帰できる（即ち、データは失われない）と想定される。|サーバーのネットワークへのアクセスの中断はサーバーの凍結の例である。|全ての所要のデータ（変数及びアルゴリズムコード）が消去又は変更されない場合でも、電力の遮断又はサーバー上の全プロセスの停止は凍結を構成する。凍結されたサーバーはメッセージを送受信せず、人の介入により再始動されるまで遊休状態である。|

・サーバーが実行するプロトコルを敵対者が変更し、最初のプロトコルに関して間違っているメッセージをサーバーが送るようにする場合、敵対者はサーバーを制御する。|他のサーバーの観点から、そのようなサーバーは不正をしている、即ち不正直である。凍結は制御の1つの

ケースである。

・サーバーが記憶している秘密データを敵対者がうまく消去又は変更したとき敵対者はサーバーを作動禁止する。プロトコルコードの消去又は他のサーバーの公開キーの消去は、システム管理がこのサーバーの活動を復帰させるのを困難にするが、この情報は公開されているので、システムオペレータに秘密情報を暴露せずに、それを再びサーバーに導入することができる。|凍結及び作動禁止の間に質的な差異を作ることはサーバーの秘密共有の破壊である。|

【0006】上記のタイプの敵対者に対する保全を提供する、秘密共有方式の望ましい特性について以下に説明する。|重要なことは、上記の基本的な秘密共有方式でのディーリング段階が保全されない状態における記憶段階及び再構築段階での保全である。|この重要性は、ディーリング段階での保全を高めることを目的とする検証可能な秘密共有方式と対照的である。

【0007】 $n$ サーバーのうちの $(k+1)$ だけが秘密の再構築のために協同する必要がある場合、秘密共有方式はしきい値方式と呼ばれる。この特性は下記のように行動する攻撃者に対してシステムを保全する：

・再構築段階中に $n$ サーバーのうちの最大 $k$ サーバーまで凍結即ち作動禁止する。|記憶段階中に、これらのサーバーが再構築段階開始前に提訴されることを条件に、敵対者はより多くのサーバーを凍結でき、しきい値方式は安全である。|

・アルゴリズムの有効期間中、 $n$ サーバーのうちの $k$ サーバーをコンプロマイズする。 $n$ サーバーのうちの $(k+1)$ サーバーをコンプロマイズすることは、敵対者が敵対者自身で秘密を再構築することを可能にする。|上記の2つの必要条件は、しきい値 $k$ が厳密に少数である、即ち次の式が成立つ場合にのみ満たすことができる。|

$$\text{【数1】 } 2k + 1 \leq n$$

【0008】最大 $k$ の不正なサーバーが存在しても再構築段階が安全である場合、秘密共有方式はロバストと呼ばれる。|それ自身によるしきい値方式は、攻撃者が作動禁止し又は凍結しうるがコンプロマイズされたサーバーを制御できないとき、秘密を再構築するために障害のない $(k+1)$ サーバーのグループを取出すのに問題がないことによる。なぜなら、障害は非活動状態を意味するからである。要するに、再構築段階で、全部で $k$ 以下のサーバーを作動禁止又は制御できる敵対者に対して、ロバストはしきい値方式を安全にする。|

【0009】しきい値秘密共有方式は、それが正しい秘密共有を失ったサーバーに復元できる場合に回復可能である。この方式は、正しい秘密共有を、それを失ったサーバーに復元できると仮定する。|従って、回復機構は記憶段階で実行される。秘密共有の回復の特徴が本発明の目的である。この特徴は、記憶段階の時点のどこかで $n$ サーバーのうちの $k$ 以下のサーバーを作動禁止又は制御

できる敵対者に対してロバストしきい値方式を安全にする。あるサーバーが作動禁止されることをシステムが知る毎に回復段階は開始し、それがあまりに短いので、この段階中に敵対者は他のサーバーに移りえないと想定すれば、それは成功である。

【0010】回復は秘密共有が失われる毎に必要であるが、それはこの秘密共有を知る敵対者がいてもなくても起こりうる。コンプロマイズ及び作動禁止能力を有する敵対者がサーバーに侵入し、その秘密共有を知り、そしてそれを消去又は変更したとき、この共有は失われる。しかし、この共有は電力遮断によっても失われる。最初のケースで、敵対者が秘密を知った直後に、その秘密が公に回復されうることが論議されることがある。しかしながら、二人以上の敵対者がおり、公開な共有回復により、仕事の一部分が他の敵対者に与えられることがある。また、電力遮断による作動禁止のケースを処理する機構が存在し、敵対者の一人が秘密を知った場合にその機構が用いられることがある。保全形式がどのケースを処理しているかを知ることは困難であるから、秘密共有が失われる毎に1つの保全回復機構を用いる方がよい。

【0011】秘密回復及びサーバー確認 敵対者が局所記憶装置及びコンプロマイズされたサーバーのメモリを消去又は変更する能力を有し、更に敵対者がサーバー間の通信チャンネルにメッセージを挿入する場合、完全な自動回復は不可能である。サーバーをリブートし、秘密共有アルゴリズムを実行するプロセスを開始し、このサーバー及び他のサーバーの間で相互確認する一定の手段を再び設けなければならない。

【0012】秘密共有に関してサーバー確認の問題が議論されるのは稀である。しかしながら、サーバー確認は、秘密共有に参加するサーバーのグループの外部からサーバーにメッセージを送りうる敵対者が存在するとき、回復プロトコルの保全を保つために必要である。サーバーAの回復は、その共有を再び設けようとするサーバーBを確認する一定の手段を持たなければならない。また、サーバーBは、それらが適切な共有を与えようとするサーバーが実際にサーバーAであることを確認しなければならない。サーバーAとサーバーBとの間の安全なリンクにより、又はAのサーバー及びBのサーバーの間の秘密／公開の暗号解読及び署名キーの対により、これらの目標を達成することができる。しかしながら、サーバーAがその秘密共有を失った場合、他のサーバーBとのリンクの保全及び確認に用いる全ての他の秘密キーも失うことがある。これは、作動禁止されたサーバーに確認の手段を設ける際の人の介入が回復のために必要であることを意味する。この介入は、サーバーAとサーバーBとの間に新しいサーバー間リンク保全を設けること、又はサーバーBの公開確認／暗号キーをサーバーAに再び設ける際にその新しい秘密キーをサーバーAに計算させてそれらを安全にサーバーBに設けることを含

む。順向システムの例で、秘密／公開キーがサーバー間通信に用いられるので、回復機構への人の介入は次のタイプである。

【0013】[順向性] 新しい秘密共有方式の特性は下記のように定義される。

順向性: 順向性 (プロアクティブ) は、敵対者がサーバーをコンプロマイズできる速度を制限することにより保全性を高める。順向秘密共有方式の記憶段階は、短い更新段階により分割されたラウンドからなる。この方式は、同じラウンドでk以下のサーバーをコンプロマイズできる敵対者に対して安全である。順向性なしに、この方式は、全記憶段階、即ち、実効的にアルゴリズムの有効期間中、最大kサーバーをコンプロマイズできる敵対者に対して安全である。順向方式は、全てのサーバーをコンプロマイズする敵対者に対して、その敵対者があまり早くコンプロマイズしないとの条件で、安全である。この特性は、更新段階の間にサーバーが保持した秘密共有を再び無作為化することにより達成される。更新プロトコルは、秘密を、プロセス中にそれを明かさずに、再デコーディングすることに等しい。形式的には、更新は下記を達成しなければならない:

(1) 更新プロトコルに参加するサーバーの数がk又はそれよりも少ないグループは他のサーバーの新しい共有について知ることはできない。

(2) 前のn共有のうちのkと新しいn共有のうちのkとを知ることは秘密共有に関する情報を明かさない。

【0014】[順向性対しきい値、ロバスト及び回復] 敵対者は、更新段階プロトコル中は再構築段階中よりも強くはならない。よって、ロバストで回復可能なしきい値秘密共有方式を順向化するために、更新段階は、最大kサーバーを作動禁止又は制御できる敵対者に対して安全でなければならない。

【0015】秘密共有での順向性及び回復可能性はそれらが一緒に用いられるとき最も意義がある。回復機構なしに、順向秘密共有は一定のタイプの攻撃に対する保全性を高めるが、他のタイプの攻撃に対しては保全性を低くする。非順向性秘密共有方式は、記憶段階の間にサーバーを制御できる敵対者に対して安全である。なぜなら、サーバーは当該段階の間には何もしないからである。順向方式の記憶段階の間に、サーバーは周期的にそれらの共有を更新する。それゆえ、サーバーは、更新段階の間に凍結又は制御される場合、次のラウンドで適切な共有を有しないであろう。これは作動禁止する能力を有する敵対者の場合に等しい。よって、順向方式は、1つのラウンドで最大kサーバーを制御できる敵対者に対して、それが回復機構を有する場合にのみ、安全である。それは、1つのラウンドで最大kサーバーを制御又は作動禁止できる敵対者に対しても安全である。

【0016】

【発明が解決しようとする課題】 本発明の第1の目的は

順向性、ロバスト及び回復可能な分散しきい値秘密共有方式を有する方法及び装置を提供することにある。〔本発明の第2の目的は上記方式を用いて順向性の安全なキー証明権限を提供することにある。〕

【0017】

〔課題を解決するための手段〕上記及びその他の目的は、サーバーのセットを用いる順向性しきい値秘密共有暗号システムを提供する方法及びシステムにより達成される。順向性秘密共有暗号システムは、 $n$ サーバーのうちの少なくとも $(k+1)$ サーバーが活動状態であり且つ正直である場合にサービスが維持される点で、分散しきい値暗号システムである。秘密署名キーは、少なくとも $(k+1)$ サーバーに敵対者が侵入する場合にのみコンプロマイズされる。それは、敵対者に侵入されたサーバーが不正をしているときでも、正直なサーバーは誤りのあるサーバーを検出し且つサービスは中断されない点でロバストである。それが回復可能であるのは、敵対者がコンプロマイズしたサーバーにある全ての局所情報を敵対者が消去する場合、サーバーは正しいプロトコルの実行に復帰すると直ちに前記情報を復元できるからである。本発明の方法及びシステムは順向性を有する。これは、敵対者が秘密を知るためには、そのアルゴリズムの同じラウンドの間に、 $(k+1)$ サーバーに侵入しなければならないことを意味する。なぜなら、秘密共有は周期的に再分散され且つ再び乱数化されるからである。2つのラウンド間の更新の間に保全必要条件を得るために、本発明は検証可能な秘密共有機構を用いる。本発明は、サーバーは同報通信媒体により通信し、それらは完全に同期され、破壊できない局所クロックを有し、そしてそれらは局所の真の乱数化のソースを有すると想定する。この方式の保全は、大きな素数順のフィールドでの対数計算の扱いにくさを想定することに依存する。それは、サーバー間の確認に用いられる「エルガマル」署名方式の保全性にも依存する。↓

【0018】

〔発明の実施の形態〕本発明はロバスト及び回復可能性を有する順向性しきい値秘密共有方式である。この方式のモデル及び目標ならびに使用するツールが開示される。本発明を使用するキー証明権限システムも開示される。この方式の保全は、大きな素数順のフィールドでの対数計算の扱いにくさを想定することに依存する。それは、サーバー間の確認に用いられる「エルガマル」署名方式の保全性にも依存する。「エルガマル」署名方式は T. ElGamal, A Public Key Cryptosystem and a Signature scheme Based on Discrete Logarithm, IEEE Trans. on Informational Theory 31, p. 465, 1985に開示されている。

【0019】〔システムのモデル及び敵対者に関する想定〕順向的に値 $x$ を秘密共有する $n$ サーバーのシステム〔数2〕 $A = \{P_1, P_2, \dots, P_n\}$

【0020】を想定する。また、このシステムは安全に且つ適切に初期化されると想定する。この方式の目標は、敵対者が $x$ を知ることを阻止すると同時に、サーバー $A$ が $x$ の再構築を必要とするとき $x$ をそれら自身で再構築するのを敵対者が阻止できないようにすることである。サーバー $A$ 及びそれらが通信する通信ネットワークに関する仕様が与えられる。また、サーバー $A$ とその人的管理の間の対話機構ならびに前記管理が負う責任も指定される。

【0021】〔順向性サーバーのモデル〕本発明を実施するサーバーの代表的なハードウェア構成は図1に示される。図1は、中央処理装置10、例えば通常のマイクロプロセッサ、及びシステムバス12を介して相互接続された複数の他の装置を有する、本発明によるワークステーションの典型的なハードウェア構成を示す。図1に示すワークステーションは、ランダムアクセスメモリ(RAM)14と、読取専用メモリ(ROM)16(図示せず)と、(ディスク装置20及びテープ装置40のような周辺装置をバス12に接続する)入出力(I/O)アダプタ18と、(キーボード24、マウス26、スピーカー28(図示せず)、マイクロホン32(図示せず)及び(又は)他のユーザインタフェース装置、例えばタッチスクリーン装置(図示せず)をバス12に接続する)ユーザインタフェースアダプタ22と、(ワークステーションを通信ネットワークに接続する)通信アダプタ34と、(バス12を表示装置38に接続する)表示アダプタ36とを含む。

【0022】図2はサーバー $A$ が秘密リンク $L$ を介して一般的な同報通信媒体 $C$ に如何に接続されるかを示す。この媒体は通信チャンネル又は通信ネットワークと呼ばれ、外部世界にも接続される。同報通信媒体は、あるサーバーにこの媒体を接続するリンクからこの媒体にメッセージが送られる毎に、この媒体に接続された全ての他のリンクにそのメッセージが瞬時に届く特性を有する。本発明は、安全で且つ同期されたクロック(図示せず)が $A$ のサーバーに装備され、敵対者がサーバーを制御又は作動禁止するときでも変更しないし、障害も起きないと想定する。これらのクロックは時間をラウンド及び更新段階に分割する。同期により、 $A$ の全てのサーバーは同時に更新段階を開始する信号を取得する。 $A$ の全てのサーバーは真に乱数化のソース(図示せず)を有することも想定される。もし敵対者がサーバーをコンプロマイズしても、その敵対者は当該サーバーに生成される将来の乱数を予測できないことも想定される。どのサーバーも、それらに関して以下に説明するアルゴリズム(図示せず)によるコードを含む比喩的な破壊不可能なボックスを有する。

【0023】〔敵対者〕サーバー $A$ を攻撃する敵対者のモデルは「移動誤り(モービル・フォルト)」モデルの拡張である。このモデルは R. Ostrovsky & M. Yung, How to Withstand Mobile Virus Attacks, Proc. of th



e 10th ACM Symposium on the Principles in Distributed Computing, 1991, pp. 51-51 and R. Canetti & A. Herzberg, Crypto 94 に記述されている。| ラウンド  $t$  の間のどこかの時点で敵対者が変造する A のサーバーの数を  $k_1(t)$  とし、ラウンド  $t$  の間のどこかの時点で敵対者による攻撃が活動状態である通信チャンネルとのリンクを有する別のサーバーの数を  $k_2(t)$  とする。そのとき、本発明の順向性秘密共有方式は各ラウンド  $t$  について

【数 3】  $(k_1(t) + k_2(t) \leq k)$

【0024】である敵対者に対して安全である。| この限界を満たすために、本発明では、A のサーバーの数  $n$  は次の式を満たす必要がある。|

【数 4】  $n \geq 2k + 2$

【0025】この方式は更に強力な敵対者に対しても安全でありうる。第 1 に、次の式に示すように、1 少ないサーバーによるラウンドで敵対者に同じ限界  $k$  を得ることができる。

【数 5】  $2k + 1 \leq n$  |

【0026】第 2 に、同期機構の変化により、リンクへの攻撃の増加がありうる。なぜなら、前記攻撃は全てシステムにより常に検出され、管理が敵対者をリンクから取り除くまでサーバーはそのプロトコルを中止しうるからである。|

【0027】サーバーの変造は下記のどれかを意味する：

- ・サーバーのコンプロマイズ：サーバーが記憶する全てのデータを知る。
- ・サーバーの制御：指定されたプロトコルに関してサーバーに誤りを生じさせる。これは機械の凍結を含む。
- ・サーバーの作動禁止：そのデータを変更（又は消去）する。|

【0028】更に、敵対者は常に公開として区分されたデータを知り、そして敵対者は機械の各々が実行するアルゴリズム  $M$  を知っている。| 簡略化のため、本発明は “普通” の電力中断、データ脱落及び他のハードウェア障害を敵対者の活動として処理する。| よって、機械に障害がある度に、それを変造したのは敵対者であると想定する。|

【0029】2 つのラウンドの間の更新段階中のどこかの時点でサーバーが変造されたとき、これらのラウンドの双方でサーバーは変造される。| 変造されたサーバーをカウントする方法の背後にある理由は、更新段階中に 1 つのサーバーから別のサーバーに移動する敵対者と、両者にずっと留まってだけいる敵対者とを識別することが不可能か又は非常に困難なことである。| 更新段階が 1 ラウンドの長さより無視できる程に短いこの設定では、この識別はあまり重要ではない。| 更に、本発明は、1 つのサーバーから他のサーバーに飛び移る敵対者を、それらのサーバーがどちらもそのラウンドの間ずつ

と変造されている場合と同じように処理する。|

【0030】A のサーバーに対する攻撃とは別に、敵対者はサーバー間の通信ネットワークを攻撃できる。敵対者は、そのリンクを介してサーバー A を接続する同じ通信チャンネル C に接続されるそれ自身のサーバー E を有すると想定する。| よって、敵対者はチャンネル C で同報通信されるメッセージを聴取し且つメッセージをチャンネル C に同報通信しうる。| しかしながら、メッセージが通信チャンネルに届くと、敵対者は、それに接続された全てのリンクに、そのメッセージが届くのを阻止できない。敵対者は同じ A のサーバーを通信チャンネルで接続するリンクへの攻撃を行うこともできる。| サーバー  $P \in A$  とのリンクが前記攻撃を受ける場合、敵対者は次の 2 つの方法で攻撃を阻止できる：1 つの方法は、チャンネル C で同報通信されるメッセージを  $P$  が聴くのを阻止することであり、もう 1 つの方法は、同報通信された  $P$  によりチャンネル C で送られるメッセージを停止することである。| 形式的には、敵対者はリンク L 及び通信チャンネル C の両者の全てのテープを見ることが出来る。| あるリンク L のアルゴリズムを敵対者のアルゴリズムと置き換えることは当該リンクへの積極的な攻撃を構成するが、チャンネル C のアルゴリズムを変更することはこのモードでは禁止される。|

【0031】本発明は、サーバー A 及びリンク L を攻撃する敵対者は “除去可能” である、即ち敵対者を、それが検出されたとき、除去できると想定する。| しかしながら、サーバー E は “除去可能” ではなく、追跡し又はチャンネル C から遮断することはできない。|

【0032】[システムの人的管理に関する想定] A の全てのサーバーは、人的管理に関する情報を表示できるコンソール 38 を有する。| マネジャは、コンソール 38 を介して、サーバーをリブートし、そして確認キーの更新の手順を実行する。コンソールプログラムによりデータがサーバーに入力される。管理は下記のように委ねられる：マネジャサーバーに干渉せず且つマネジャはマネジャのために指定されたプロトコルで不正をしない。| 換言すれば、管理はサーバーをリブートし且つ確認キーの再設定の時期及び方法に関する指示に任される。|

【0033】人的管理はリンク上の攻撃する敵対者を突き止める責任を有する。| リンクが攻撃されているかどうかを検査し且つ潜在的な攻撃者を除去するように管理が指示されるときは必ず、手順は常に成功するものと仮定する。| また、サーバーをリブートし、公開確認キーを再設定し、そしてリンクから敵対者を除去する全ての手順に要する時間は 1 ラウンドの長さよりも短いと想定する。|

【0034】[秘密共有方式] 図 3 の流れ図は本発明の順向性検査可能秘密共有方式の良好な方法を示す。図 4 は図 3 のステップ 314 の詳細な流れ図である。同様に、図 5 は図 4 のステップ 410 の詳細な流れ図である。|

【0035】[初期化] ステップ310で、サーバーは初期化される。 $p$ 及び $q$ は素数とし、 $m$ は2、3又は4のような小さな整数とし、次の式が成立つものとする。

$$\text{【数6】 } p = mq + 1 \quad |$$

【0036】 $g$ は次数 $q$ の $Z_p$ の元素である、即ち次の式が成立つものとする。

$$\text{【数7】 } g^q = 1 \pmod{p} \quad |$$

【0037】素数 $p$ は“エルガマル”暗号化及び署名方式を安全にするために選択される。秘密値 $x$ は $Z_q$ に属する。限定フィールド $Z_q$ にわたる“シャミール”の秘密共有の変形をしきい値秘密共有方式として使用する。 $Z_q$ に $(k+1)$ 次の多項式 $f$ が存在し、次の式が成立つものとする。

$$\text{【数8】 } f(0) = x \pmod{q} \quad |$$

【0038】そして、全てのサーバー $P_i, i \in \{1, n\}$ は次の式で示されたその秘密共有を有するものとする。

$$\text{【数9】 } x_i(1) = f(i) \pmod{q} \quad |$$

【0039】指数(1)はこれらが最初のラウンドで用いた値になることを示す。

【0040】更に、各サーバー $P_i$ は、その秘密確認及び暗号化キー $w_i(1)$ 乱数を $Z_q$ 内に有する。それらのキーの公開の対応部は次の式に示される。

$$\text{【数10】 } \{r_i(1)\} \quad i \in \{1, n\} \quad |$$

【0041】ここで、次の式は公開であり且つAの全てのサーバーにより記憶される。

$$\text{【数11】} \quad |$$

$$r_i(1) = g^{w_i(1)} \pmod{p} \quad |$$

【0042】更に、あらゆるサーバーは秘密を再構築する際のロバストについて全ての $x$ の一方方向ハッシュのセットを必要とする。本発明は一方方向ハッシュとして累乗を用いるので、あらゆるサーバーは次の式で示されたセットを記憶する。

$$\text{【数12】 } \{y_i(1)\} \quad i \in \{1, n\} \quad |$$

【0043】ここで、次の式が成立つ。

$$\text{【数13】} \quad |$$

$$y_i(1) = g^{x_i(1)} \pmod{p} \quad |$$

【0044】モデルの説明で述べたように、本発明は初期化段階では(受動的にも能動的にも)敵対者が存在しないと想定するので、この初期化は公然と行いうる。

【0045】初期化の後、Aのサーバーの局所クロックは時を刻み始める。この時点で、複数のサーバーの動作は離散した時間間隔、即ちラウンドで同期される(ステップ312)。各ラウンドの終りで、更新段階がトリガーされる(ステップ314)。更新段階で、サーバーAは更新プロトコルを実行する。そして管理が任意の(最大 $k$ まで

の)キー再設置手順とそれに続く共有再構築プロトコルを実行するために予約された時間がある。更新段階はこれらの全てを達成する十分な長さを有するが、ラウンドと比較すれば短いと想定される。

【0046】[更新プロトコル] 共有を更新するために、ステップ314及び図4に示すように、本発明は“アール・オストロフスキー”及び“エム・ユング”が提案した更新プロトコルの簡略化されたバージョンに適合する。 $Z_q$ にある $k$ 次の多項式の値 $f(0)$ が $x$ に等しい場合に秘密 $x$ が記憶されるとき、秘密 $x$ はそれを $k$ 次の無作為の多項式 $\delta(\cdot)$ に加えることにより更新できる。ここで、 $\delta(0)$ は0に等しいので、次の式が成立つ。

$$\text{【数14】} \quad |$$

$$f^{(t+1)}(0) = f^{(t)}(0) + \delta(0) = x + 0 = x \quad |$$

【0047】“オストロフスキー”及び“ユング”は、ある点で多項式を計算する演算の直線性が次の式の非常に簡単な共有更新ができることに注目した。

$$\text{【数15】 } x_i = f(i) \quad |$$

【0048】即ち、次の関係式が成立つ。

$$\text{【数16】} \quad |$$

$$f^{(t+1)}(\cdot) = f^{(t)}(\cdot) + \delta(\cdot) \pmod{q} \quad \langle====\rangle$$

$$\forall i \quad f^{(t+1)}(i) = f^{(t)}(i) + \delta(i) \pmod{q} \quad |$$

【0049】本発明では、次の式が成立つ。

$$\text{【数17】} \quad |$$

$$\delta(\cdot) = (f_1(\cdot) + f_2(\cdot) + \dots + f_n(\cdot)) \pmod{q} \quad |$$

【0050】次の式で示される $(k+1)$ 次の多項式は、 $i$ 番目のサーバーにより独立して且つ無作為に取り上げられる。

$$\text{【数18】 } f_i(\cdot), f_i(0) = 0, i \in \{1, n\} \quad |$$

【0051】各サーバー $P_i$ の更新プロトコル、 $i \in \{1, n\}$ は下記ようになる：

(1)  $P_i$ は次の式で示される $(k+1)$ の乱数を $Z_q$ から取出す。

$$\text{【数19】 } \{f_{ij}\} \quad j \in \{1, (k+1)\} \quad |$$

【0052】これらの数は次の式に示す多項式を定義する。

$$\text{【数20】} \quad |$$

$$f_i(z) = f_{i1}z^1 + f_{i2}z^2 + \dots + f_{i(k+1)}z^{(k+1)} \quad |$$

【0053】その自由な係数は0であるので、次の式が成立つ。

$$\text{【数21】 } f_i(0) = 0 \quad |$$

【0054】(2) 全ての他のサーバー $P_j$ に関して、 $P_i$ は次の式を $P_j$ に送る。

$$\text{【数22】 } f_i(j) \pmod{q} \quad |$$

【0055】(3)  $P_i$ は次の式で示されるその新しい共有を計算する。

$$\text{【数23】} \quad |$$



$$x_i^{(t+1)} \leftarrow x_i^{(t)} + (f_1(i) + f_2(i) + \dots + f_n(i)) \pmod{q}$$

【0056】そして $P_i$ は、その現在の秘密キー  $x_i^{(t+1)}$ を除き、それが用いた全ての変数を消去する。

【0057】制御能力を有する敵対者に対してこのプロトコルを安全にするために、本発明は P. Feldman, A Practical Scheme for Non-Interactive Verifiable Secret Sharing, Proc. of the 28th IEEE Symposium on the Foundations of Computer Science, pp.427-37. 1987 で提案され、そして T.P. Pederson, Distributed Provers with Applications to Undeniable Signature, Euro crypto '91, 1991で論議された一方向機能を用いて検査できる秘密共有の機構を用いる。この特定の検査可能秘密共有方式が用いられるのは、それが非対話型であり、そしてその副次効果として、それが  $x_i$  の更新と一緒に秘密の指数  $y_i$  の更新を可能にするからである。

【0058】下記の2つの場合に、正直なサーバーは、サーバー  $P_i$  により生成された更新多項式  $f_i$  を、それらが共有することを異議なく "無効" とマークすべきである。

- (1)  $P_j$  が受取る共有  $f_i(j)$  が  $(k+1)$  次の多項式の値ではない、又はそれらはその値であるが、この多項式  $f_i$  は、 $f_i(0) \neq 0$  であるので、正しい更新多項式ではない。
- (2) 公開値  $y_j$  の更新に用いられる共有

【数24】

$$\{g^{-f_i(j)}\}_{j \in \{1, n\}}$$

【0059】が、 $f_i$  の秘密更新共有

【数25】  $\{f_i(j)\}_{j \in \{1, n\}}$

【0060】に対応しない。

【0061】秘密/公開キー対応部としての値  $w_i/r_i$  は、 $P_i$  から来るメッセージの確認、及び "エルガマル" 暗号化による  $P_i$  宛のメッセージの暗号化に用いられる。 $P_i$  が  $P_j$  について  $m \in Z_q$  を暗号化するとき、それは

【数26】  $E_j^k[m] = (m(r_j)^k, g^k)$  ?

【0062】を送る。ここで、 $k \in Z_q$  は乱数であり、そして指数は  $Z_p$  で計算される。受信装置は

【数27】

$$m = m(r_j)^k * (g^k)^{(-w_i)} \pmod{p}$$

$$(g^{f_{i1}}, g^{f_{i2}}, \dots, g^{f_{i(k+1)}}) \cdot E_1^{k_{i1}} [U_{i1}] \cdot E_2^{k_{i2}} [U_{i2}], \dots, E_{(i-1)}^{k_{i(i-1)}} [U_{i(i-1)}] \cdot E_{(i-1)}^{k_{i(i+1)}} [U_{(i+1)}], \dots, E_n^{k_{in}} [U_{in}] \quad (1)$$

【0074】を生成する。ここで、全ての指数が  $Z_p$  で計算される (ステップ514)。更に、各サーバー  $P_i$  は、 $Z_q$  内

【0063】を用いて解読する。

【0064】署名動作は衝突のないハッシュ機能  $h: N \rightarrow Z_q$  を用いる。キー  $w_i$  を有するメッセージ  $m$  の署名は、

【数28】  $S_i[h(m)] = (r, s)$

【0065】である場合に、

【数29】

$r = g^k \pmod{p}$ ,  $s = k^{-1} (h(m) - rw_i) \pmod{q}$ ,

【0066】である。この署名は、 $w_i$  の公開対応部

【数30】

$$r_i = g^{w_i} \pmod{p}$$

【0067】により、等式

【数31】

$$g^{h(m)} \stackrel{?}{=} r^s r_i^r \pmod{p}$$

【0068】を検査することにより確認できる。ハッシュ機能  $h$  は、 $m' \neq m$  の場合に誰もいかなる対  $m'$ ,  $S_i[h(m')]$  を生成できない  $m$ ,  $S_i[h(m)]$  を知る特性を有しなければならない。

【0069】全更新段階を通じて、暗号化及び確認は値  $\{w_i(t), r_i(t)\}$  により実行される。ここで  $t$  は終了したばかりのラウンドである。このラウンド中にサーバーをコンプロマイズさせた敵対者は、このサーバーが更新プロトコルの間に署名及び解読に用いる秘密キーを知るであろう。

【0070】[完全な更新プロトコル] 下記のステップはステップ314でサーバーが用いた更新プロトコルの詳細である。

【0071】ステップ410で、各サーバー  $P_i$  は、 $Z_q$  で更新多項式

【数32】  $f_i(z) = f_{i1}z^1 + f_{i2}z^2 + \dots + f_{i(k+1)}z^{k+1}$

【0072】を定義する  $Z_q$  にある  $(k+1)$  の乱数  $f_{i1}, f_{i2}, \dots, f_{i(k+1)}$  を取出す (ステップ510)。それは、この多項式の  $(n-1)$  の共有

【数33】  $h_{ij} = f_i(j) \pmod{q}$ ,  $j \neq i$

【0073】を形成し (ステップ512)、そしてそのメッセージ  $msg_i$

【数34】

でその新しい秘密キー  $w_i^{(t+1)}$  を乱数として取出し (ステップ516)、そしてその対応する新しい公開キー:

【数35】

$$r_i^{(t+1)} = g^{w_i^{(t+1)}} \pmod{p}$$

【0075】を計算する(ステップ518)。そして $P_i$ は、その古いキー $w_i(t)$ により署名された対:

【数36】  $(msg_i, r_i^{(t+1)})$ 

【0076】を同報通信する(ステップ412 及び520)。

【0077】ステップ414で、各サーバー $P_i$ は、それが前のステップで受信したメッセージを考慮する。ある $j$ で、それが形式

【数37】  $(msg_j, r_j^{(t+1)})$ 

$$g^{u_{ij}} = (g^{f_{j1}})^i (g^{f_{j2}})^{i^2} \dots (g^{f_{j(k+1)}})^{i^{k+1}} \pmod{p} \quad (2)$$

【0080】の

【数40】

$$(g^{f_{j1}}, g^{f_{j2}}, \dots, g^{f_{j(k+1)}})$$

【0081】部分で与えられた係数に $u_{ij}$ が一致するかどうかを確認する。

【0082】この式が適用できない場合、 $P_i$ は $P_j$ を“無効”とマークし、 $B_j$ から $j$ を除去し、そしてそれが $P_j$ の不正であったことに対する告発:

【数41】  $acc_{ij} = (i, j)$ 

【0083】を作成する。サーバーがマークした“無効”の各々について、 $P_i$ は、管理への対応するメッセージをその表示装置に表示し、それがなぜ $P_i$ の何かが無効であるかを示す。そして、ステップ416で、サーバーは、それが前のステップで受信した全ての新しい公開キー $\{r_j\}_{j \in B}$ のセットと連結された全てのその告発の署名されたセットを同報通信する。署名は古い秘密キー $w_i(t)$ により行われる。

【0084】ステップ418で、各サーバー $P_i$ は前のステップで同報通信されたメッセージの署名を検査する。2つ以上のメッセージが同じ署名で同報通信される場合、対応するサーバーは“無効”とマークされ、セット $B_i$ は変えられ、そして管理のための適切なメッセージが $P_i$ のコンソールに表示される。サーバーは、新しい公開キー $r_i^{(t+1)}$ が $P_j$ により肯定応答されなかったように、対のセット $(j, 1)$ をそれらのコンソールに表示する。これ

$$E_j^{k_{ij}}[u_{ij}] \neq (u_{ij}^{(r_j(t))})^{k_{ij}} \pmod{p}, g^{k_{ij}} \pmod{p}$$

【0088】であれば、即ち $P_i$ が $P_j$ の告発に全く応答しなかったとき、 $P_i$ は不正をしているので、それは“無効”とマークされなければならない。さもないと、(b) もし $(u_{ij}, k_{ij})$ が実際に $msg_j$ で用いられたならば、ステップ(2)で $P_j$ が用いるべき同じ等式(2)を評価することにより、

【0078】の確認されたメッセージを受信しないか又は2つ以上受信した場合、それは $P_j$ を“無効”とマークする。 $B_i$ は“無効”とマークされなかったサーバーの索引のセットとする。そして $P_i$ はメッセージ $msg_j, j \in B_i$ の

【数38】

$$E_i^{k_{ji}}[u_{ji}]$$

【0079】部分を解読する。そして、あらゆる $j \neq i$ について、それはメッセージ $msg_j$ :

【数39】

はリンク攻撃が活動状態である場合に管理に追跡させる。しかしながら、新しい公開キーの肯定応答の欠如は $B_i$ に影響しない。あるサーバー $P_j, j \in B_i$ による告発 $acc_{ji}$ の各々について、サーバー $P_i$ は、 $P_j$ との通信で用いられた共有及び乱数ベクトルを含む署名された応答:

【数42】  $resp_{ij} = (i, j, k_{ij}, u_{ij})$ 

【0085】を同報通信するので、誰が不正をしていたかを決定するための公開審査を可能にする。これらの応答は1つの署名と連結され、同報通信されるべきである。

【0086】ステップ420で、同じサーバーの署名を有する2つ以上の応答が同報通信された場合、全てのサーバー $P$ はそれを“無効”とマークし、そのセット $B_i$ を変え、そして管理のメッセージを表示する。各サーバー $P$ は、ステップ(1)で $P_i$ により適切に同報通信された $(acc_{ji}, resp_{ij})$ の全ての対についてそれ自身の決定を行う。resp<sub>ij</sub>で送られた値は疑わしいとみなされるので、それらは次の式に示す“素数”により表示される:

【数43】

$$k_{ij}^{(r_j(t))}, u_{ij}^{(r_j(t))}$$

【0087】あらゆるサーバー $P$ は下記のアルゴリズムにより決定する:

(a) もし

【数44】

$$u_{ij}^{(r_j(t))} = u_{ij}$$

【数45】

【0089】は多項式 $f_i$ の正しい共有であるかどうかを検査する。この等式が正しくない場合、それは $P_i$ が無効な共有を $P_j$ に送ったことの証明であるので、それは“無

効”とマークされなければならない。等式が正しい場合、PはP<sub>j</sub>を”無効”とマークする。

【0090】”無効”とマークされた各サーバーについて、セットB<sub>i</sub>は変えられ、そして対応するメッセージが

$$x_i^{(t+1)} \leftarrow x_i^{(t)} + \sum_{k \in B_i} [f_k(1)] \pmod{q}$$

$$\forall j \neq i \left[ y_j^{(t+1)} \leftarrow y_j^{(t)} * \prod_{k \in B_i} g_k^{f_k(j)} \pmod{p} \right]$$

【0092】各サーバーは新しい共有  $x_i^{(t+1)}$ 、新しいキー  $w_i^{(t+1)}$ 、そして新しい公開キーのセット

【数47】  $\{y_j^{(t+1)}\} \quad j \in \{1, n\}$

【0093】及びセット

【数48】  $\{r_j^{(t+1)}\} \quad j \in \{1, n\}$

【0094】を除いて、このプロトコルで用いた全ての変数を除去する。

【0095】正しい更新多項式  $\delta(\cdot)$  は合計

【数49】

$$\sum_{k \in B} [f_k(\cdot)] \pmod{q}$$

【0096】に等しい。ここで、少なくとも  $(k+2)$  のサーバーP<sub>i</sub>,  $i \in \{1, n\}$  について  $B = B_i$  である (ステップ420)。

【0097】サーバーPが既知の形式のメッセージの署名された同報通信の受信を予期しており、そしてそれが2つの前記メッセージを受信し、どちらも同じサーバーP<sub>i</sub>から来るらしいときは必ず、PはP<sub>i</sub>を無効とマークする。もちろん、攻撃者はP<sub>i</sub>が行ったものと同じメッセージを常に送信できる。このプロトコルでは、全てのサーバーは最大3つのメッセージまで送信する。それゆえ、Pは、P<sub>i</sub>が署名したように見えるメッセージのセットC<sub>i</sub>を受取るときは必ず、該プロトコルのどれか前のステップでP<sub>i</sub>により同報通信されたC<sub>i</sub>の要素を捨てるべきであり、そしてC<sub>i</sub>の全ての他の要素が同じである場合、この要素をP<sub>i</sub>のメッセージとみなす。さもなければ、P<sub>i</sub>を”無効”とマークすべきである。なぜなら、それは不正をしているか、又はその確認キーをコンプロマイズした敵対者により複製されているからである。別の良好な点は、P<sub>i</sub>がそれ自身が署名したように見えるメッセージを調べる場合、それはそれ自身も”無効”とマークし、従ってB<sub>i</sub>を計算することである。このように、たとえそれが複製されていても、それはその新しい共有  $x_i^{(t+1)}$  及び全ての

【数50】  $\{y_j^{(t+1)}\} \quad j \in \{1, n\}$

【0098】を正しく計算する。その新しい公開確認及び暗号化キー  $r_i^{(t+1)}$  は肯定応答されないで、気を付けなければならない。

【0099】[更新プロトコルの分析] 敵対者は次の3つの方法で上記プロトコルに干渉することができる：

P<sub>i</sub>のコンソールに表示される。

【0091】ステップ422で、各サーバーP<sub>i</sub>は次の式を計算する：

【数46】

・ 敵対者はこのプロトコルでその敵対者が管理するサーバーに不正をすることができる。使用禁止されたサーバーは（それが間違ったキーでメッセージを暗号化し且つ署名する場合）それが不正をしているかのように見えるか又はそれが乱数メッセージを送っているかのように見える。

・ 敵対者はその敵対者がこの更新段階前のラウンド中にコンプロマイズしたサーバーのキーにより署名されたメッセージを送信できる。他のサーバーについては、このケースはサーバーがなお敵対者により制御されるときの場合と区別できない。

・ 敵対者はサーバー及び通信チャンネルの間のリンクを攻撃できる。

【0100】全ての更新ラウンドで、正直であり、前のラウンドでコンプロマイズされておらず、通信チャンネルとのリンクに対する攻撃は活動状態ではない、 $(K+2)$ のサーバーがあるから、これらのサーバーのセットB<sub>i</sub>は全て同じであり、そして少なくとも $(k+2)$ の要素を有する。それゆえ、秘密共有多項式  $f$  は少なくとも $(k+2)$ の多項式  $f_i$  により更新され、敵対者はそれらをどれも再構築できない。最悪の場合、敵対者はこれらの多項式の各々から  $k$  の共有を学習できるが、それらは $(k+1)$ 次の全てであり且つそれらの各々について  $f_i(0) = 0$  であるから、敵対者はそれらを再構築するためには $(k+1)$ の共有を必要とする。

【0101】[更新後の回復] サーバーのコンソールに表示されたメッセージから、管理は、敵対者によりどの機械が制御され、コンプロマイズされ又は使用禁止され、そしてどのリンクに対する攻撃が活動状態であったかを決定できる。特に、コンソールは下記の2つのセットの（しばしば交差する）サーバーに情報を表示する：

・ 第1のセットはそれらの新しい公開キーの幾つかの肯定応答を受信しなかったサーバーである。

・ 第2のセットはそれらの新しい公開キーが少なくとも $(k+2)$ のサーバーにより肯定応答されず、それらの秘密共有  $x_i^{(t+1)}$  が間違っただけで計算され、そして新しい多項式  $f(t+2)$  と一致せず、更に、次の式に示すそれらの現在のセットのビューが正しくないサーバーである。

【数51】  $\{y_j^{(t+1)}\} \quad j \in \{1, n\}$

【0102】第1のセットは活動状態のリンク攻撃に関

する管理情報を与える。そしてマネジャは導き出されたリンクから迅速に敵対者を取り除く。第2のセットに関して、マネジャは最初にキー再設定手順(ステップ316)を実行し、そして自動共有回復プロトコル(ステップ318)を開始させる。公開キー $\{r_j\}$ が良好に分散され且つ他のサーバーにより肯定応答されているAのサーバーのセットをCとする。そして次の式が成立つものとする。

$$【数52】 |C| \geq k+2$$

【0103】  $P_i$ のキーの再設定(ステップ316)は次に行われる： $P_i$ のコンソールにより、マネジャは $\{r_j\}_{j \in C}$ を設定し、そして $P_i$ にその新しい乱数の秘密キー $w_i \in Z_q$ を取出すように求める。 $P_i$ はこのような数を取出し、そして次の式に示す対応する公開キーを表示する。

【数53】

$$r_i = g^{w_i} \pmod{p}$$

【0104】そしてマネジャは全ての他のサーバーにこの数を設定する。セットCは $P_i$ により増加される。この再設定が終了すると、マネジャはサーバーCに関する回復プロトコルを開始させる。あるいは、サーバー自身は、ある公開キーがそれらに再設定されるときは必ず、回復プロトコルを自動的にトリガーするためにそれらが同意プロトコルを開始するようにプログラムされる。そのとき、Cにある全てのサーバーは使用可能である。

【0105】【共有回復プロトコル】  $P_u$ は再設定を必要とする、次の式の共有を有するサーバーを示す。

$$【数54】 x_u = f(u)$$

【0106】最初に、ステップ610で、サーバーは $(k+2)$ のサーバーの、次の式の初期セットを選択する。

$$【数55】 B \subset C \setminus \{P_u\}$$

【0107】それらは $P_u$ に対する $q$ 自身の確認又はその逆の確認をしなければならないのでCから取出される。これは容易である。なぜなら、サーバーは索引により順序付けられ、そしてそれらはCの現在の状態、即ち動作中の確認及び暗号化キーを有するサーバーを知っているからである。該プロトコルは敵対者が存在するとき下記の特性を有しなければならない：

- ・  $P_u$ は $x_u$ のみを学習する、即ち、 $i \in B$ である場合、他の共有 $x_i$ を学習できない。

- ・  $i \in A \setminus F$ である場合、 $(k-1)$ のサーバーのグループ

$$【数56】 F \subset A \setminus \{P_u\}$$

【0108】は $x_u$ を又はいかなる $x_i$ も学習できない。

- ・ サーバー $P_u$ は適切なセット $\{y_i\}_{i \in \{1, n\}}$ を学習する。

【0109】 $f$ の有効な共有を有する $(k+2)$ のサーバーの基本セットは

$$【数57】 x = f(0) \pmod{q}$$

である場合だけでなく、 $f$ の任意の他の値、特に $P_u$ が必

要とする共有

$$【数58】 x_u = f(u) \pmod{q}$$

【0110】も回復できる。これはラグランジェ補間法の公式で行うことができる。協同することになっている $(k+2)$ のサーバーのセットBが設定された後、各 $P_i$ ,  $i \in B$ は $P_u$ に

【数59】

$$a_i = x_i \prod_{j \in B, j \neq i} [(u-j)/(i-j)] \pmod{q}$$

【0111】を送る。そして $P_u$ はそれらを加え合わせて

【数60】

$$x_u = f(u) = \sum_{j \in B} a_j$$

【0112】を取得する。しかしながら、本発明がそれのみを行った場合、 $P_u$ は全ての $x_i$ を $a_i$ からも学習するであろう。しかし、本発明は $a_i$ を $x_v$ の些細な秘密共有として処理できる。それゆえ、本発明はBのサーバーがこれらの共有を再乱数化した後にそれらを $P_u$ に送ることを必要とする。Bの各サーバー $P_i$ は $Z_q$ にある $(k+2)$ の乱数 $\{c_{ij}\}_{j \in B}$ を取出す。そして、それらはこれらの値を対で交換する： $P_i$ は $c_{ij}$ を $P_j$ に与え、そして $c_{ij}$ を取得する。次に、それぞれが

【数61】

$$a_i' = a_i + \sum_{j \in B} c_{ji} - \sum_{j \in B} c_{ji} \pmod{q}$$

【0113】を $P_u$ に送る。

【数62】

$$\sum_{i \in B} a_i' = x_u \pmod{q}$$

【0114】であるから、これは $x_u$ のもう1つの秘密共有である。ここで基本方式は更新段階プロトコルで用いた同じツールを用いて検査できなければならない。

【0115】この方式を検査可能にする完全なプロトコルを以下に記述する。全ての同報通信は、現時点でセットBにないものを含むCのサーバー及びサーバー $P_u$ を必要とする。明確化を図るために、下記は、全ての確認された同報通信について、受信者はメッセージの署名を検査し、そして(送信者が不正をしているか又はそのキーがコンプロマイズされ複製されているために)複製の攻撃があるとき、それらはこのサーバーを“無効”とマークし、それらの管理のコンソールに適切なメッセージを表示することに言及しない。これは更新プロトコルと同じ手順である。この方式を検査可能にするプロトコルは：Bの全ての $P_i$ が $Z_q$ にある $(k+2)$ の乱数値 $\{c_{ij}\}_{j \in B}$ を計算し、そして：

【数63】

$$(\{g^{c_{ij}} \pmod{p}\}_{j \in B}, \{E_j^{k_{ij}}[C_{ij}]\}_{j \in B}) \quad (3)$$

【0116】及びこのメッセージの署名を同報通信する(ステップ612)。

【0117】ステップ614で、Bの全ての $P_i$ は上記同報通信から値  $\{c_{ji}\}_{j \in B}$  を解読し、そしてそれらが $P_j$ により同じメッセージで同報通信された指数

【数64】

$$g^{c_{ji}}$$

【0118】を取ることににより、全てのサーバー $P_j$ から正しい共有を与えられたかどうかを検査する。この値が一致しない場合、 $P_i$ は $P_j$ の告発

【数65】  $acc_{ij} = (i, j, S_i[i, j])$

【0119】を全てのサーバーに同報通信する（ステップ616）。それは $P_j$ も“無効”サーバーとマークする。

【0120】ステップ618で、更新プロトコルと全く同じように、あるサーバー $P_j$ による告発 $acc_{ji}$ 毎に、サーバー $P_i$ は署名  $S_j[j, i]$  を検査し、そして告発の原告が  $x$ を知っていることが証明された場合、 $P_i$ は、 $P_j$ との通信で用いられた $c_{ij}$ 及び乱数ベクトル:

【数66】  $resp_{ij} = (i, j, k_{ij}, c_{ij})$

【0121】に回答し、誰が不正をしていたかを決定する公開審理を可能にする（ステップ620）。|

【0122】ステップ622で、(Bのサーバーのみならず)全てのサーバーPは、2つの前のステップで同報通信された全ての対  $acc_{ji}, resp_{ij}$  について、これらが  $P_i$ 及び $P_j$ の間の通信で用いられた真の値であるかどうかを検査し、そしてそれらが真である場合、送られた値の指数

【数67】

$$g^{c_{ji}} \pmod{p}$$

【0123】がそのセットにある $P_i$ により同報通信された指数

【数68】

$$\{g^{c_{ij}} \pmod{p}\}_{j \in B}$$

【0124】と一致するかどうかを検査する。

【0125】このステップは更新プロトコルのステップ420での検査に等しい。 $P_i$ 又は $P_j$ が不正をしていると分かり、そしてPにより“無効”とマークされる。

【0126】ステップ624で、あらゆるサーバーは、それが前のステップで“無効”とマークしたその局所セット $F_i$ のサーバーを考慮する。

【数69】  $|F_i| = m \neq 0$

【0127】である場合、それらは新しいB:

【数70】

$$B \leftarrow (B \setminus F) \cup \{\max(B)+1, \dots, \max(B)+m\}$$

【0128】を計算する。同報通信により、正直で活動状態の全てのサーバーは同じビューのFを有する、従って同じビューの新しいBを有する。それらはBを再計算したのちステップ610でプロトコルを再開する。

【0129】ステップ626で、前のステップで誰も不正

をしなかった場合、全てのサーバー $P_i$ はその下位共有:

【数71】

$$a_i' = a_i + \sum_{j \in B} c_{ij} - \sum_{j \in B} c_{ji} \pmod{q} \quad (4)$$

【0130】を計算する。ステップ628で、Bの各サーバー $P_i$ は全てのサーバーに

【数72】

$$E_{r_u}^{k_i} [a_i']$$

【0131】を同報通信する。ここで、 $k_i$ は $Z_q$ にある乱数である。

【0132】ステップ630で、 $P_u$ はこれらの同報通信から全ての

【数73】

$$\{a_i'\}_{j \in B}$$

【0133】を解読し、そして

【数74】

$$x_u = \sum_{i \in B} a_i' \pmod{q}$$

【0134】を計算する。そして、それは、この値の指数

【数75】

$$g^{x_u} \pmod{p}$$

【0135】が、最初に $P_u$ で設定された公開キー  $y_u$ と同じであるかどうかを検査する。同じである場合、再構築はステップ632で終了する。さもなければ、Bからのサーバーのなかには不正をしていたものがあるに違いない。Pは全ての  $i \in B$  について:

【数76】

$$g^{a_i'} \leftarrow (y_i)^{\prod_{j \in B, j \neq i} [(v-j)/(i-j)]} \pmod{p}$$

$$g^{a_i'} = g^{a_i} * \prod_{j \in B} g^{c_{ij}} * \left( \prod_{j \in B} g^{c_{ji}} \right)^{-1} \pmod{p}$$

【0136】を評価することにより不正をした者を見つける。ここで、値

【数77】

$$\{g^{c_{ij}}\}_{i, j \in B}$$

【0137】はステップ612で同報通信から取出される。2番目の式が適用できない場合、それは $P_i$ が正しい  $a_i'$  を  $P_u$ に送らなかったことを意味する。それを他に對して証明するために、 $P_u$ はその告発

【数78】  $acc_i = (i, S_{temp}[i])$

【0138】を全ての他のサーバーに同報通信する(ス

テップ634)。また、それは不正をしたものをそれ自身で全て“無効”セットFとマークする。

【0139】ステップ636で、このプロトコルのステップ618におけるように、全ての告発されたサーバー $P_i$ は、告発 $acc_i$ の署名を検査し、もしそれが有効であれば、次の式に示す応答を同報通信する。

【数79】

$$resp_i = (i, k_i, a_i)$$

【0140】ステップ638で、全てのサーバーPは全てのこれらの告発を検査するに当り、最初にこれらが通信で用いた値であることを検査し、次に上記の実行された $P_u$ と同じ検査式を検査する。更に、それらは全てそれらが無効であると検出した同じサーバーのセットFで到来する。不正をしているのは $P_u$ であることが判明した場合、それらは全て適切なメッセージをシステムマネージャに送り、そして回復手順は停止される(ステップ640)。この事象は $P_u$ が敵対者により再び制御されることを意味する。これは他の敵対者検出と全く同様である。システムマネージャはサーバーが表示した警告を読み取り、そして全回復プロセスを再開する。しかしながら、 $P_u$ の告発が正しい場合、通常のように、サーバーはシステム管理のための適切な警告を表示し、そしてセットB:

【数80】

$$B \leftarrow (B \setminus F) \cup \{\max(B) + 1, \dots, \max(B) + m\}$$

【0141】を再計算することにより回復を続ける。ここで、

【数81】 $|F| = m$

であり、そしてステップ610から全プロトコルを再開する。

【0142】[制御されたサーバーが存在するときの更新プロトコルの保全] 不正をする $k$ のサーバーが存在するときの上記プロトコルの保全及びリンクに対する攻撃がないことは、検査可能な秘密共有に変えられる。即ち、解決は、どれか特定のサーバー $P_i$ がそのプロトコルで不正をしていたかどうかを正直なサーバー $P_j$ が検出できる機構である。このケースでは、それらは異議なくその更新多項式 $f_i$ の共有を“無効”とマークする。上記プロトコルで $P_i$ が不正を行うのは、それがステップ612で他のサーバーに送る値が $Z_q$ で、

【数82】 $f(0) = 0 \pmod{q}$

が成立つ、任意の $k$ 次の多項式 $f$ の正しい値ではない場合、そしてその場合のみである。これは、検査された値の共有と呼ばれる、検査可能な秘密共有の幾分か簡略化されたケースである。なぜなら、全てのサーバーPは $(n-1)$ の他のサーバーの間に既知の値0を分散する(それはそれ自身にも1つの共有を与える)。検査可能な秘密共有と全く同じように、正直なサーバーは、“秘密の”値0がPにより正しく共有されたかどうか異議なく同

意するであろう。また、検査可能な秘密共有と全く同じように、それは、各サーバーがそれ自身の秘密共有のみを学習するように行われるべきである。

【0143】変更された更新プロトコルのステップ618~620は上記から:

【数83】

$$x_i^{(t+1)} \leftarrow x_i^{(t)} + \sum_{k \in B} [f_k(i)] \pmod{q}$$

【0144】である。ここで、 $B_i$ は、更新多項式 $P_i$ により“無効”とマークされなかった全てのサーバーの索引のセットである。検査可能な秘密共有機構により、更新段階の間に $P_i$ 及び $P_j$ がどちらも正直である場合、

【数84】 $B_i = B_j$

【0145】である。即ち、他のサーバーの正直さについてそれらは同じ判断を有する。

【0146】既存の検査可能な秘密共有のプロトコルは全て同報通信チャンネルを必要とする。よって、この点から、 $n$ の秘密共有サーバーの各々はそれを普通の同報通信チャンネル、エサネットに接続するリンクを有すると想定する。この同報通信チャンネルに乗る全てのメッセージは、このチャンネルをサーバーに接続する全てのリンクに到着する。

【0147】更新の間に最大 $k$ の不正をしている、私用禁止、即ち凍結されたサーバーが存在するときの検査可能な秘密共有が達成される場合、全ての正直なサーバーがその共有を更新する、少なくとも $(n-k)$ の更新多項式 $f_i$ がある。

【0148】共有の再乱数化の次数は順向方式の第2の特性、即ち更新前及び更新後に $k$ の共有を知ることは敵対者に秘密を再構築させないことを保証するのに十分である。以下は $k$ サーバーが更新中に制御されるとき微妙な点である。敵対者はそのとき各更新多項式 $f_i$ の $k$ の共有を知り、そしてこれは敵対者がそれらの全てを再構築することを許すのは、それらの全ての自由な係数が0であることが知られているからである。よって、敵対者は全てのサーバー $P_i$ の全更新値

【数85】

$$\delta(i) = (f_1(i) + f_2(i) + \dots + f_n(i)) \pmod{q}$$

【0149】を計算できる。しかしながら、更新段階の間に敵対者が $k$ サーバーのセットFをコンプロマイズする場合、敵対者はこの更新の前又は後のラウンドで他のサーバーをどれもコンプロマイズすることができない。よって、敵対者は全ての

【数86】 $\{x_i(t)\} \quad i \in F$

【0150】及び

【数87】 $\{\delta(i)\} \quad i \in \{1, n\}$

【0151】を知る。後者は、特に敵対者が、どのみち知る

【数88】 $\{x_i(t+1)\} \quad i \in F$



【0152】を計算することを許す。しかし、秘密の再構築を敵対者に許す追加の共有

【数89】

$$x_i(t), j \notin F$$

【0153】又は

【数90】

$$x_i(t+1), j \notin F$$

【0154】を敵対者は依然として学習できない。

【0155】["重複"攻撃及びリンク攻撃が存在するときの更新プロトコルの保全] 敵対者がラウンド $t$ でサーバー $A$ をコンプロマイズし、そして同じラウンドでサーバー $A$ から去る場合、ラウンド $t$ 及びラウンド $(t+1)$ の間の更新段階中に、このサーバーは無効ではないが、敵対者は全てのその秘密を知る。(活動状態のプロトコルの混乱がない) コンプロマイズされたサーバーのみを有する敵対者を想定する。ラウンド $t$ の間及び更新 $t/(t+1)$ の間に敵対者によりコンプロマイズされたサーバーのセットを $F$ とする。最悪のケースでは、

【数91】 $|F| = (k-1)$

である。そのラウンドの間及び更新の間に敵対者に対して絶対に安全である"有効な"サーバーのセットを $G$ とする。明らかに、それらがサーバー $F$ に送る各 $f_i$ は、 $i \in G$ である。さて、プロトコルにより、それは、 $A$ が取得する $f_i$ 、 $i \in G$ の共有を学習することもあり、しないこともある。サーバーが既知のキーで暗号化されたばかりのそれらの更新多項式の共有を $A$ に送る場合、敵対者はそれらを学習できる。

【0156】[順向性安全公開キー証明権限] 順向性安全キー証明権限センターを実現するために順向性秘密共有を用いる方法を示す完全な解決が提案される。メッセージの"エルガマル"署名動作を実行し且つ署名キーを順向的に維持するシステムが記述される。このようなシ

$$y \leftarrow y_1 y_2 \dots y_n = g^{x_1} g^{x_2} \dots g^{x_n} = g^{x_1 + x_2 + \dots + x_n} \pmod{p} \quad (5)$$

【0161】を計算できる。シーケンス $(p, g, y, n)$ はこの署名センターで生成された署名の検査に用いる公開キーである。

【0162】[署名の発行] メッセージ $m$ に署名するために、各サーバーはそれ自身の秘密乱数 $k_i$ 、 $(p-1)$ に関連する素数を取り出す。そして、各サーバー $P_i$ はその署名の第1の部分:

【数95】

$$r_i = g^{k_i} \pmod{p}$$

【0163】を計算する。各サーバーはその $r_i$ を他のサーバーに同報通信するので、それらの各々は、次の式に示すように、その署名の第2の部分の計算する:

$$S(m) = (m, s_1, s_2, \dots, s_n, r_1, r_2, \dots, r_n) \quad (7)$$

システムはキー証明権限として有効に用いうる。なぜなら、それが署名するメッセージは公開キー及び他のユーザの識別タグにありうるからである。

【0157】[エルガマル署名の分散バージョン] 最初に、互いに署名センターを形成する参加サーバーのグループにより署名動作が実行される、"エルガマル"署名アルゴリズムの分散バージョンを想定する。署名はこれらのサーバーの部分的な署名から成る。それは1つの公開検査キーにより検査できる。これは、概念的に、単一の署名キーもあることを意味し、それは署名センターを形成するサーバー間で共有されるだけである。このアルゴリズムは、それを共有するサーバー間のこの秘密キーの分散の変化を許す、即ち、それは共有の順向性更新を許す。

【0158】[初期化] 大きな素数を $p$ とし、そして $p$ よりも小さい乱数を $g$ とする。 $p$ 及び $g$ はどちらもネットワーク内の全ての当事者が知っている。各サーバー $P_i$ はその秘密キーを乱数 $x_i$ とみなす。ここで、 $i \in \{0, n\}$  且つ  $x_i < p-1$ である。全てのサーバーの秘密キーが初期化された後、その結果生ずる全てのラウンドで、次の式の合計が得られる。

$$【数92】 x_1(t) + x_2(t) + \dots + x_n(t) = x \pmod{q}$$

【0159】ここで、 $x_i(t)$ はラウンド $t$ でのサーバー $P_i$ の秘密キーを表わし、そして $x$ は署名センターの概念的な一定の秘密キーを表わす。最初に、各サーバーはその秘密キーの公開対応部:

【数93】

$$y_i = g^{x_i} \pmod{p}$$

【0160】を計算する。そして、それらはそれらの公開キーの部分を互いに送出するので、 $x$ の公開検査対応部:

【数94】

【数96】

$$s_i = k_i^{-1} (m - x_1 r_1 r_2 \dots r_n) \pmod{p-1} \quad (6)$$

【0164】最初の"エルガマル"署名方式と全く同じように、

【数97】

$$k_i^{-1} \pmod{p-1}$$

【0165】の値は、ユークリッドのアルゴリズムにより見出だすことができる。署名されたメッセージは下記のシーケンスを有する:

【数98】

【0166】 [署名の検査] 署名の検査を必要とする当事者は下記の式が真であるかどうかを検査する:

$$g^m = y^{r_1 r_2 \dots r_n} r_1^{s_1} r_2^{s_2} \dots r_n^{s_n} \pmod{p} \quad (8)$$

【0167】 これは下記の変換により真であると示すことができる:

$$\begin{aligned} y^{r_1 \dots r_n} r_1^{s_1} \dots r_n^{s_n} &= (g^{x_1 + \dots + x_n})^{r_1 \dots r_n} (g^{k_1})^{k_1^{-1} (m - x_1 r_1 \dots r_n)} \dots \\ (g^{k_n})^{k_n^{-1} (m - x_n r_1 \dots r_n)} &= g^{x_1 r_1 \dots r_n + \dots + x_n r_1 \dots r_n} g^{m - x_1 r_1 r_2 \dots r_n} \\ * g^{m - x_n r_1 \dots r_n} &= g^{n \neq m} \pmod{p} \end{aligned}$$

【0168】 [保全分析] 上記の基本的な分散 "エルガマル" アルゴリズムの保全は、本発明のプロトコルの残りの部分と同様に、大きな素数順の最後のフィールドにおける対数の計算は見込まれた多項式の計算時間から実行不可能であるとの想定による。これは "エルガマル" 及びDSA暗号及び署名方式の基本的な保全を想定する。上記の "エルガマル" の公開キーアルゴリズムの分散バージョンの保全は最初のバージョンと全く同じである。攻撃者が本発明のアルゴリズムを破壊できる場合、攻撃者は通常の "エルガマル" アルゴリズムも破壊できるであろう。攻撃者は  $m$ 、 $S(m)$  及び  $x_1, x_2, \dots, x_{n-1}$  を知

$$g^x = (g^{x_n})^{s_n} \pmod{p}, \quad g^k = (g^{k_n})^{r_1 r_2 \dots r_n} \pmod{p}$$

【0171】 及び

【数104】

$$x + 1 = s_n k_n + x_n r_1 r_2 \dots r_n = m \pmod{p-1}$$

【0172】 を知る。この3つの式のセットで、最初の2つのうちの1つは残りの部分から取出すことができる。例えば、

$$【数105】 g^k = g^{x+k} (g^x)^{-1} \pmod{p-1}$$

【0173】 である。よって、残るのは下記の2つの式である。

【数106】

$$g^x = a \pmod{p}, \quad x + k = b \pmod{p-1}$$

【0174】 最初の "エルガマル" アルゴリズムの保全は、 $k$  が未知である場合、第2の式は  $x$  に関する情報を出さないことに基づくので、"エルガマル" アルゴリズムは見込まれる多項式の計算時間で対数

$$【数107】 x = \log_a \pmod{p}$$

【0175】 を計算することは実行不可能であるのと同程度に安全である。

【0176】 [順向性しきい値秘密共有方式への分散エルガマルの適応] 本発明は "シャミール" の多項式秘密共有を上記分散 "エルガマル" アルゴリズムに組込む。この組込みはしきい値順向方式を生成するが、このしきい値はラウンド中にのみ存在する。前述のように、更新段階中のしきい値は回復機構を必要とする。この方式は

【数99】

【数100】

っていると想定する。攻撃者が更に必要とするものは  $x_n$  である。攻撃者は

【数101】

$$g^{x_n} = y * g^{-(x_1 + x_2 + \dots + x_{n-1})}$$

【0169】 を計算できる。ここで、負の指数はモジュロ  $(p-1)$  が計算される。

$$【数102】 x = x_n s_n, \quad k = k_n r_1 r_2 \dots r_n$$

【0170】 が代入される。そして攻撃者は

【数103】

ロバストを達成しない。よって、この予備的な方式は:

- ・各ラウンドで最大  $k$  サーバーをコンプロマイズでき、
- ・各ラウンドで最大  $k$  サーバーを凍結するが、更新段階では凍結できない敵対者に対して安全である。

【0177】 本発明は、Threshold Cryptosystems Crypto to 89, pp. 307-15 に記述された、指数のしきい値秘密共有に関する "デスメド" 及び "フランケル" の解決に関連する。彼らの着想は、 $k$  次の多項式による "シャミール" のしきい値秘密共有を用い、そして各サーバーで局所的にラグランジェの補間式の成分を計算することにより  $(k+2)$  の正直なサーバーのグループによる秘密の指数を計算することであった。それらの方式及び本発明の間の類似性は、多項式により共有された秘密のラグランジェ再構築を両者が用いることであるが、プロセス中に再構築の秘密を誰も学習できない。秘密は ("デスメド" 及び "フランケル" のケースでは) メッセージを指数化するか又は (本発明では) "エルガマル" により取出された署名を発行するためにのみ用いられる。本発明は  $n$  サーバー間で  $f(0) = x$  を秘密共有するために  $k$  次の多項式  $f$  を用いる。それは  $p$  及びエレメント  $g \in \mathbb{Z}_p$  を取出し、 $g$  が素数順序  $q$  になるようにする、即ち

$$【数108】 g^q = 1 \pmod{p}$$

【0178】 である。 $Q$  はできるだけ大きくすべきであり、

【数109】  $p = mq + 1, m \in \{2, 3, 4\}$

【0179】が成立つようにする。ここで、公開キーはシーケンス  $(p, q, g, y, k)$  になる。

【0180】本発明は、初期化段階の間に、各サーバーがその  $x$  の秘密共有

【数110】  $x_i = f(i)$

$$a_i = x_i * \prod_{j \in B, j \neq i} [(0-j)/(i-j)] \pmod{q} \quad (9)$$

【0182】を計算する。ラグランジェの補間公式から、これは

【数112】

$$\sum_{i \in B} a_i = x \pmod{q} \quad (10)$$

【0183】であることを保証する。なぜなら、

【数113】  $\forall i, x_i = f(i), x = f(0)$

【0184】であり且つ  $f$  は  $Z_q$  で  $k$  次の多項式であるからである。

【数114】  $(i-j), i \neq j$

の逆数の存在は、 $q$  が素数であることにより保証される。それらの部分的な秘密として  $a_i$  を用いることにより、 $B$  のサーバーは、上記：

【数115】

$$g^{(k+2)*m} = y \prod_{(r_i, s_i) \in S(m)} r_i \prod_{(r_i, s_i) \in S(m)} r_i^{s_i} \pmod{p} \quad (11)$$

【0187】【署名プロトコルにおける検査及びロバスト】 上記予備的なしきい値方式に付加しなければならない第1の特性はラウンドの間のロバストである。これは、最大  $k$  サーバーまで制御できる敵対者に対してそれを安全にするが、それはそのラウンドの間のみであって、更新段階の間ではない。上記しきい値方式は署名者従属である。なぜなら、署名を発行する前に、正直なサーバーは、そのメッセージに署名するために協同する、 $(k+2)$  のサーバーのセット  $B$  に同意する必要があるからである。幾つかの現に活動状態である、正直なサーバーのセット  $B$  を取出しうするためには、正直なサーバーが不正をする者を取り除きうる機構が必要である。最後に、 $B, m$  及び  $S(m)$  を知る各サーバーは、全てのサーバー  $P_i, i \in B$  について、同じ  $m, g, p, q$  及び式(5)により計算された正しい秘密キー  $a_i$  を有するプロトコルにより計算されるという意味で、その部分的な署名  $(r_i, s_i) \in S(m)$  が正しいかどうかを検査しうるべきである。

【0188】サーバー間のこの互いに部分的な検査を可

$$a_i = (y_i)^{\prod_{j \in B, j \neq i} [(0-j)/(i-j)]} \pmod{q} \pmod{p}$$

【0192】を計算する。

(2) 上記のように

【数121】

【0181】を取得すると想定する。ここで、 $f$  は  $Z_q$  で計算される。署名を発行するために、サーバーはどれか  $(k+2)$  の活動状態のサーバーのグループ  $B$  で署名に参加することに同意する。他のサーバーは遊休状態である。セット  $B$  を知り、各サーバー  $P_i, i \in B$  は：

【数111】

$$r_i = g^{k_i} \pmod{p}$$

$$s_i = k_i^{-1} * (m - a_i \prod_{j \in B} r_j) \pmod{q}$$

$$S(m) = (m, \{(r_i, s_i)\}_{i \in B})$$

【0185】からの基本的な順向性プロトコルに従って、それらの部分的な署名  $(r_i, s_i), i \in B$  を発行する。

【0186】検査式は次の式になる：

【数116】

$$g^{(k+2)*m} = y \prod_{(r_i, s_i) \in S(m)} r_i \prod_{(r_i, s_i) \in S(m)} r_i^{s_i} \pmod{p} \quad (11)$$

能にするために、本発明は各サーバーの秘密共有  $x_i(t)$  をそれらの秘密署名キーとして処理し、そして全ての他のサーバーにそれらのキーの公開検査対応部：

【数117】

$$y_i(t) = g^{x_i(t)} \pmod{p}$$

【0189】を与える。全ての正直なプロセッサ  $P$  (現に  $B$  にあるものだけではない) は  $B$  の全てのサーバー  $P_i$  の部分的な署名：

【数118】  $(r_i, s_i) \in S(m)$

【0190】を次の2つのステップで検査する。

(i) 現在の  $B$  及び

【数119】

$$y_i = g^{x_i} \pmod{p}$$

【0191】を知り、 $P$  は

【数120】

$$g^{a_i}$$

【0193】を計算したのち、 $P$  は  $S(m)$  を取出し、そ

して式:

$$g^m = (g^{a_1})^{\prod (r_j, s_j)} \in S(m)^{r_1} * r_1^{s_1} = g^{a_1 \prod (r_j, s_j)} \in S(m)^{r_j} * (g^{k_1})^{k_1^{(-1)}} * (m - a_1 \prod (r_j, s_j)) \in S(m)^{r_j} \pmod{p}, c \quad (12)$$

【0194】により $P_i$ の署名を検査する。

【0195】このように、全ての正直なサーバーは、署名  $S(m)$  を発行している間に不正が行われた無効なサーバーのセット  $F \subset B$  を決定できる。サーバーはそれらの索引  $i \in \{1, n\}$  により順序付けられるから、正直なサーバーが検出された不正な者を取り除くためにセット  $B$  を更新し、そして新しい活動状態のサーバーのセットを試

$$B \leftarrow (B \setminus F) \cup \{\max(B) + 1, \dots, \max(B) + m\} \quad (13)$$

を取る。

【0197】本発明は、全てのラウンドで最大  $k$  のサーバーが敵対者に乗っ取られて無効となりうるから、後に続く更新は（不正をする者が “1対1” で現われる場合には） $k$  回よりも多くは実行されないと想定する。これはラウンド毎に最大  $k$  の余分な署名を発行するのに等しい。これは無視できるオーバーヘッドである。

【0198】上記からの更新プロトコルは、ラウンド  $t$  毎に、秘密共有  $x_i(t)$  の公開対応部  $y_i(t)$  の維持を保証

$$y_i^{(t+1)} \leftarrow y_i^{(t)} * (g^{f_1(i)} g^{f_2(i)} \dots g^{f_n(i)}) \pmod{p} \quad (14)$$

【0200】 $Z_p$  の対数を計算する “難しさ” を仮定すれば、

【数128】

$$g^{f_1(j)}$$

【0201】の同報通信は敵対者を支援しない。

【0202】まとめとして、本発明の構成に関して以下の事項を開示する。

(1) 傾向性、ロバスト及び回復可能な分散しきい値秘密共有を有する公開キー暗号化の方法であって、キーを形成するために通信ネットワークによりリンクされたサーバーを初期化するステップと、終了を有する別個のラウンドで動作するように前記サーバーを同期させるステップと、前記通信ネットワークで同報通信されたメッセージから前記ラウンドの前記終了で更新されたキーを計算するステップと、コンプロマイズされたサーバーのセットを形成するために前記更新されたキーを検査するステップと、前記コンプロマイズされたサーバーのセットを回復するステップとを含む方法。

(2) 前記初期化するステップは、前記サーバーの各々の乱数を選択するステップと、前記乱数から前記サーバーの各々の秘密値を計算するステップと、前記秘密値から前記サーバーの各々の秘密キーを計算するステップと、前記秘密キーの公開対応部を前記通信ネットワークで同報通信するステップとを含む、上記(1)に記載の方

すことは自明である。例えば、それらが

【数123】 $B = 1, 2, \dots, k + 2$  により開始する全てのラウンドの始めに、そしてそれらが署名して  $m > 0$  の不正な者のセット

【数124】 $F = \{i_1, i_2, \dots, i_m\}$

【0196】を検出する毎に、それらは

【数125】

するように拡張できる。ステップ(2)で、サーバー  $P_i$  が  $f_i(j)$  を  $P_j$  に送るとき、それは

【数126】

$$g^{f_1(j)}$$

【0199】を全ての他のサーバーにも送る。これは全てのサーバーが次の式のように全ての  $i$  の公開キーを計算することを可能にする。

【数127】

法。

(3) 前記更新されたキーを計算するステップは、前記サーバーの各々の多項式を定義する乱数のセットを取出すステップと、前記多項式から導き出された、前記サーバーの各々の新しい秘密キーを取出すステップと、前記多項式から導き出されたメッセージを前記通信ネットワークで同報通信するステップとを含む、上記(1)に記載の方法。

(4) 前記更新されたキーを検査する前記ステップは、無効なサーバーのセットを形成するために前記通信ネットワークで同報通信された前記メッセージを分析するステップと、前記無効なサーバーのセットに対応する告発のセットを生成するステップと、前記告発のセットを前記通信ネットワークで同報通信するステップと、更新された無効なサーバーのセットを形成する前記同報通信された告発のセットを分析するステップとを含む、上記(1)に記載の方法。

(5) 前記コンプロマイズされたサーバーのセットを識別するメッセージをコンソールに表示するステップを更に含む、上記(1)に記載の方法。

(6) 前記サーバーのセットを回復するステップは、前記コンプロマイズされたサーバーのセットにある各サーバーに新しい秘密キーを設定するステップと、前記サーバーから回復サーバーのセットを選択するステップと、前記回復サーバーのセットにある各サーバーの下位共有

を計算するステップと、前記下位共有から取出されたメッセージを前記通信ネットワークで同報通信するステップと、前記コンプロマイズされたサーバーのセットが受取った前記下位共有から取出された前記メッセージを検査するステップとを含む、上記(1)に記載の方法。

(7) 回復サーバーのセットを選択する前記ステップは、前記サーバーの下位セットを選択するステップと、前記下位セットの各サーバーの乱数値のセットを計算するステップと、前記乱数値のセットから取出された署名されたメッセージを前記通信ネットワークで同報通信するステップと、無効サーバーのセットを取出すために前記署名されたメッセージを検査するステップと、前記サーバーの前記下位セットから前記無効サーバーのセットを除去するステップとを含む、上記(6)に記載の方法。

(8) 前記サーバーの各々の乱数を取出すステップと、前記乱数から前記サーバー毎の第1の署名の部分の計算するステップと、前記第1の署名の部分を前記通信ネットワークで同報通信するステップと、前記第1の署名の部分から前記サーバーの各々の第2の署名の部分の計算するステップとを更に含む、上記(1)に記載の方法。

(9) 署名されたメッセージを前記第2の署名の部分により検査するステップを更に含む、上記(8)に記載の方法。

(10) 順向性、ロバスト及び回復可能な分散しきい値秘密共有を有する公開キー暗号方式を処理するデータ処理システムであって、通信ネットワークによりリンクされたサーバーと、前記サーバーと関連したキーを形成するために前記サーバーを初期化する初期化手段と、終了を有する別個のラウンドに前記サーバーの動作を同期させるタイミング手段と、更新されたキーを生成するために前記別個のラウンドの各ラウンドの終了で前記キーを更新する更新手段と、コンプロマイズされたサーバーのセットを形成するために前記更新されたキーを検査する検査手段と、前記コンプロマイズされたサーバーのセットを回復する回復手段とを備えるデータ処理システム。

(11) 前記サーバーの各々の乱数を選択する選択手段と、前記乱数から前記サーバーの各々の秘密値を計算する第1の計算手段と、前記秘密値から前記サーバーの各々の秘密キー及び前記秘密キーの公開対応部を計算する第2の計算手段と、前記秘密キーの各々の前記公開対応部を前記通信ネットワークで同報通信する同報通信手段とを備える、上記(10)に記載のデータ処理システム。

(12) 前記更新手段は、前記サーバーの各々に関連した、多項式を定義する乱数のセットを生成する乱数生成手段と、前記多項式から前記サーバーの各々の新しい秘密キーを生成する秘密キー生成手段と、前記通信ネットワークで前記多項式から取出されたメッセージを同報通信する同報通信手段とを備える、上記(10)に記載のデータ処理システム。

(13) 前記検査手段は、無効なサーバーのセットを形成するために前記通信ネットワークで同報通信されたメッセージを分析する第1の分析手段と、前記無効なサーバーのセットに対応する告発のセットを生成する告発手段と、前記告発のセットを前記通信ネットワークで同報通信する同報通信手段と、無効なサーバーの更新されたセットを形成するために前記同報通信された告発のセットを分析する第2の分析手段とを備える、上記(10)に記載のデータ処理システム。

(14) 前記コンプロマイズされたサーバーのセットを表示する表示手段を更に備える、上記(10)に記載のデータ処理システム。

(15) 前記回復手段は、前記サーバーから回復サーバーのセットを選択する選択手段と、前記回復サーバーのセットにあるサーバーの各々の下位共有を計算する計算手段と、前記回復サーバーのセットにあるサーバーの各々の前記下位共有から取出されたメッセージを前記通信ネットワークで同報通信する同報通信手段と、前記コンプロマイズされたサーバーのセットが受取った前記下位共有の各々から取出された前記メッセージを検査する検査手段とを更に備える、上記(10)に記載のデータ処理システム。

(16) 前記選択手段は、前記サーバーの下位セットを選択する選択手段と、前記下位セットにある各サーバーの乱数値のセットを計算する計算手段と、前記乱数値のセットから取出された署名されたメッセージを前記通信ネットワークで同報通信する同報通信手段と、無効なサーバーのセットを取出すために前記署名されたメッセージを検査する検査手段と、前記無効なサーバーのセットを前記下位セットから除去する除去手段とを備える、上記(15)に記載のデータ処理システム。

(17) 前記サーバーの各々の乱数のセットを生成する乱数生成手段と、前記乱数のセットから前記サーバーの各々の署名の第1の部分の計算する第1の計算手段と、前記署名の第1の部分を前記通信ネットワークで同報通信する同報通信手段と、前記第1の署名の部分から前記サーバーの各々の署名の第2の部分の計算する第2の計算手段とを更に備える、上記(10)に記載のデータ処理システム。

(18) 署名されたメッセージを前記第2の署名の部分により検査する検査手段を更に備える、上記(17)に記載データ処理手段。↓

【図面の簡単な説明】

【図1】サーバーコンソールを示す図である。

【図2】通信チャンネルに接続するサーバーを示すブロック図である。

【図3】本発明の良好な実施例の高レベルのステップのシーケンスを示す流れ図である。

【図4】本発明の共有更新プロトコルの良好な実施例のステップのシーケンスを示す流れ図である。

【図5】図4のステップ410におけるステップのシーケンスを示す流れ図である。

【図6】本発明の共有回復プロトコルの良好な実施例のステップのシーケンスを示す流れ図である。

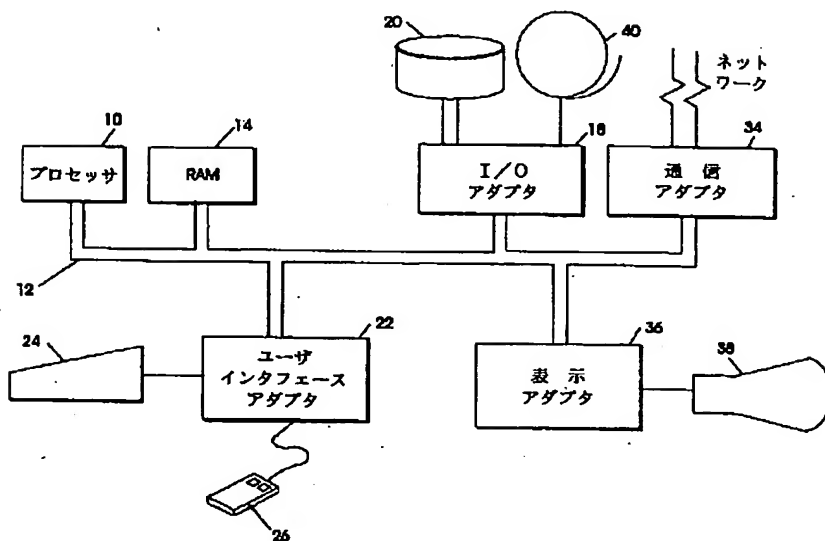
【図7】本発明の共有回復プロトコルの良好な実施例のステップのシーケンスを示す流れ図である。

【符号の説明】

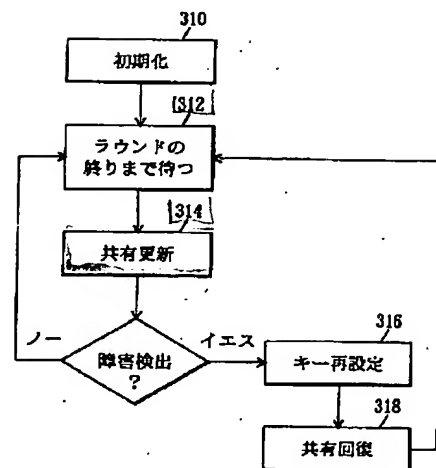
10 中央処理装置  
12 システムバス  
14 ランダムアクセスメモリ(RAM)

18 入出力(I/O) アダプタ  
20 ディスク装置  
22 ユーザインタフェースアダプタ  
24 キーボード  
26 マウス  
34 通信アダプタ  
36 表示アダプタ  
38 表示装置  
40 テープ装置

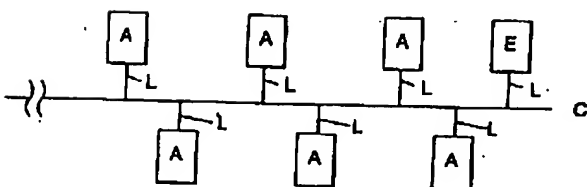
【図1】



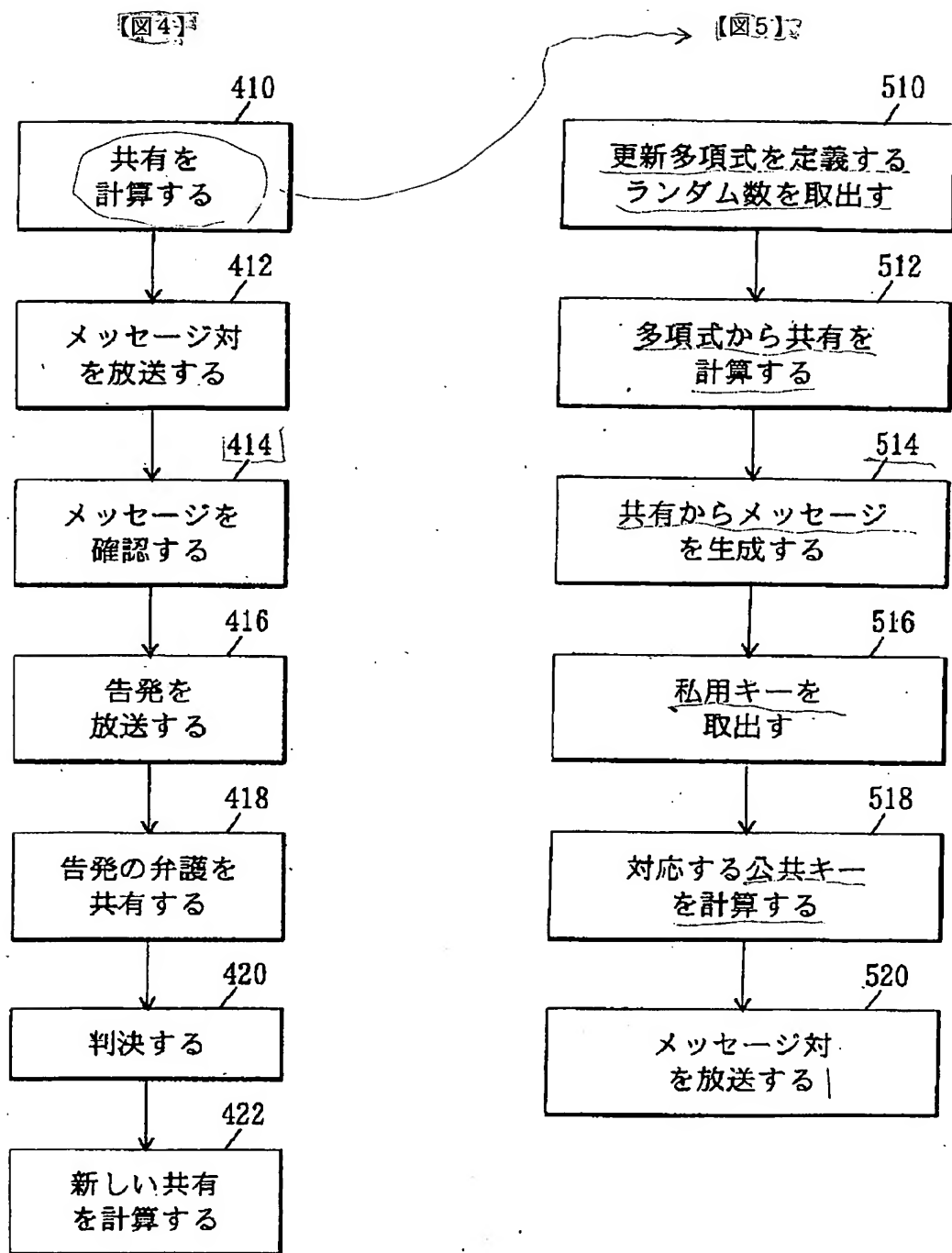
【図3】



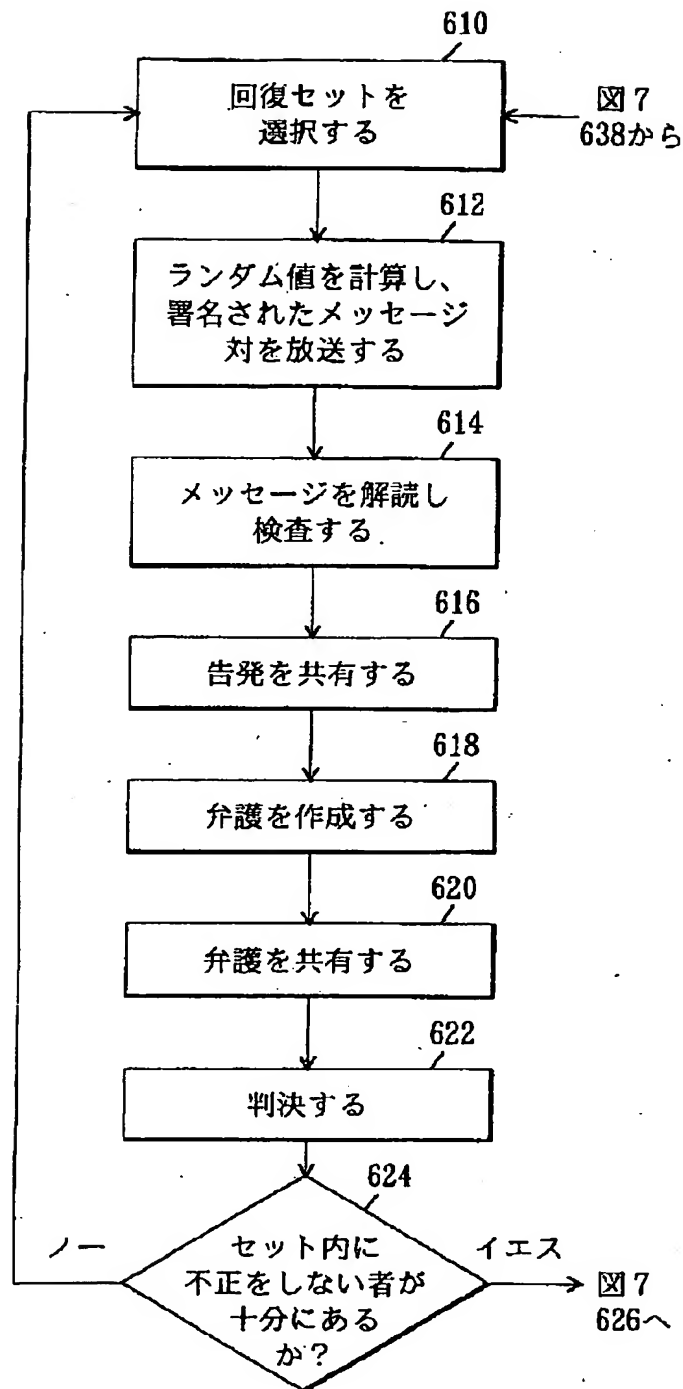
【図2】



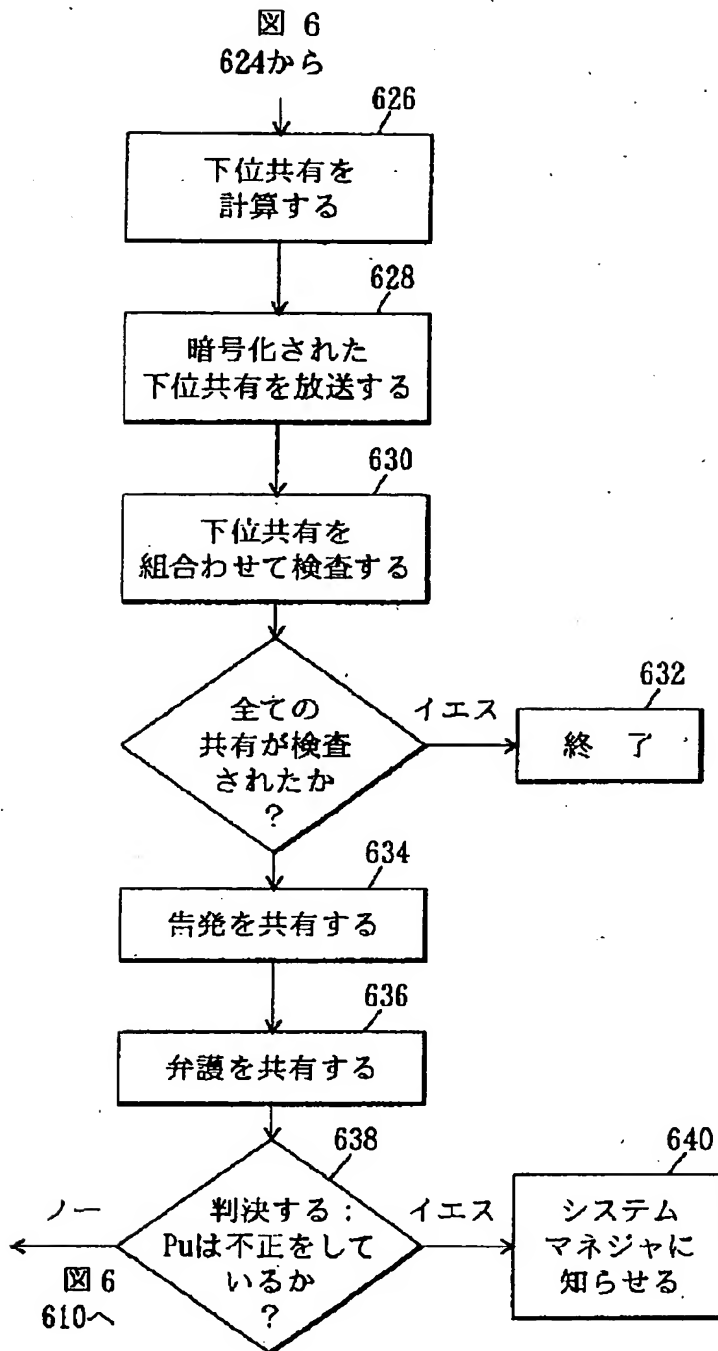




【図6】



【図7】



フロントページの続き

(72)発明者   スタニスロー・マイケル・ヤレッキー  
              アメリカ合衆国マサチューセッツ州、ケン  
              ブリッジ、マサチューセッツ・アベニュー  
              282、アパートメント・423・ディ

(72)発明者   ヒューゴ・マリオ・クロウジック  
              アメリカ合衆国ニューヨーク州、リバーデ  
              イル、ネザーランド・アベニュー   2600

(72)発明者   マーセル・モードチャイ・ユング  
              アメリカ合衆国ニューヨーク州、ニューヨ  
              ーク、ワシントン・ストリート   1

**This Page is Inserted by IFW Indexing and Scanning  
Operations and is not part of the Official Record**

**BEST AVAILABLE IMAGES**

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ **BLACK BORDERS**
- ☐ **IMAGE CUT OFF AT TOP, BOTTOM OR SIDES**
- ☐ **FADED TEXT OR DRAWING**
- ☒ **BLURRED OR ILLEGIBLE TEXT OR DRAWING**
- ☐ **SKEWED/SLANTED IMAGES**
- ☐ **COLOR OR BLACK AND WHITE PHOTOGRAPHS**
- ☐ **GRAY SCALE DOCUMENTS**
- ☐ **LINES OR MARKS ON ORIGINAL DOCUMENT**
- ☐ **REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY**
- ☐ **OTHER:** \_\_\_\_\_

**IMAGES ARE BEST AVAILABLE COPY.**

**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**